REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* 08-09-2006	**2. REPORT TYPE** Conference Proceedings	**3. DATES COVERED *(From – To)*** 26 July 2006 - 28 July 2006

4. TITLE AND SUBTITLE

Explorations in the Complexity of Possible Life

Proceedings of the 7th German Workshop on Artificial Life

5a. CONTRACT NUMBER FA8655-06-1-5017

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

S. Artmann, P. Dittrich (Eds.)

5d. PROJECT NUMBER

5d. TASK NUMBER

5e. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Friedrich-Schiller-University
Zwaetzengasse 9
Jena 07737
Germany

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

EOARD
PSC 821 BOX 14
FPO 09421-0014

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
CSP 06-5017

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited. (approval given by local Public Affairs Office) Copyrighted Material

13. SUPPLEMENTARY NOTES

Copyright 2006 Akademische Verlagsgesellschaft Aka GmbH, Berlin, Germany.
The Department of Defense has permission to use for government purposes only. All other rights are reserved by the copyright holder.

14. ABSTRACT

Topics of the conference will cover the whole area of research in Artificial Life (AL), in particular:

Modelling of biological processes, systems biology, adaptive behaviour, pattern recognition, perception and evolution of sensors, morphology and actuators, learning and intelligence in biological and artificial systems, complexity and its emergence in biological processes, self-organization in living and life-like systems (e.g. swarms, hypercycles, multicellular systems, artificial multi-agent systems), dynamics of information in living and life-like systems, application of principles of life (e.g. self-organization, evolution, adaptation, learning and intelligence) to the design of artificial systems and technical solutions (e.g. robots, hardware, and software), and epistemological foundations for sciences investigating living systems.

15. SUBJECT TERMS
EOARD, Biotechnology, Artificial Intelligence

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18, NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT UNCLAS	**b. ABSTRACT** UNCLAS	**c. THIS PAGE** UNCLAS	UL	156	ROBERT N. KANG, Lt Col, USAF
					19b. TELEPHONE NUMBER *(Include area code)* +44 (0)20 7514 4437

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39-18

S. Artmann, P. Dittrich (Eds.)

Explorations in the Complexity of Possible Life

Abstracting and Synthesizing the Principles of Living Systems

Proceedings of the 7th German Workshop on Artificial Life
July 26-28, 2006
Jena, Germany

GWAL-7

IOS Press

infix

Explorations in the Complexity of Possible Life Since its inception in the 1980s, Artificial Life is an interdisciplinary field of science focused on abstracting the essential features and dynamics of living systems in order to create artificial life-like and living systems. In 1995, the first German Workshop on Artificial Life (GWAL) was organized by young researchers who were interested in this newly emerging science and wanted to discuss their projects and results in an open, informal atmosphere. The 7th GWAL, which was held in Jena 2006, continued the successful series of these workshops. It was attended by young scientists not only from Germany but also from several other European countries and from Japan. The submissions were intensely reviewed by renowned members of an international program committee. The finally accepted contributions cover many different topics of Artificial Life, such as information and category theoretical models of living systems, the computational relevance of the physical substrate of life, the logic of genetic regulatory networks, and new approaches to artificial chemistries.

Both editors of the volume do research in Artificial Life at the Friedrich-Schiller-University Jena but have different scientific backgrounds. Stefan Artmann is a philosopher of science who is working on general problems of applying information theory and semiotics in biology. He has published epistemological papers about Artificial Life as a structural science. Peter Dittrich is a computer scientist and leader of the Biosystem Analysis Group at the Institute of Informatics. He has published extensively about artificial chemistries and network theory.

IOS Press

infix

ISBN 3-89838-076-9 (Aka)
ISBN 1-58603-644-0 (IOS Press)

S. Artmann, P. Dittrich (Eds.)

Explorations in the Complexity of Possible Life

Abstracting and Synthesizing the Principles of Living Systems

Proceedings of the 7th German Workshop on Artificial Life
July 26–28, 2006
Jena, Germany

AKA

Bibliographic information published by *Die Deutsche Bibliothek*
Die Deutsche Bibliothek lists this publication in the *Deutsche Nationalbibliografie*; detailed bibliographic data is available on the Internet at http://dnb.ddb.de

Publisher: Akademische Verlagsgesellschaft Aka GmbH, Berlin
www.aka-verlag.de

"infix" is a joint imprint of Akademische Verlagsgesellschaft Aka GmbH (Berlin) and IOS Press BV (Amsterdam)

Reproduced from electronic file supplied by the Editors
Printing and Binding: Hundt Druck GmbH, Köln
Printed in Germany

ISBN 3-89838-076-9 (Aka)
ISBN 1-58603-644-0 (IOS Press)

Contents

Committees vii

Sponsors viii

Artificial? Life?
A Kind of Preface to Any Possible Exploration in the Complexity of Life-as-It-Could-Be
Stefan Artmann and Peter Dittrich 1

Invited Talks 6

Automated Design of Artificial Life Forms from Scratch
Mark Bedau 6

The Emergence and Collapse of Complexity: Perspective from an Evolving Network Model
Sanjay Jain 7

Goedel Machines
Jürgen Schmidhuber 8

General Concepts 9

Information and Closure in Systems Theory
Nils Bertschinger, Eckehard Olbrich, Nihat Ay, and Jürgen Jost 9

An Approach to Algorithmic Music Composition with an Artificial Chemistry
Tomoya Miura and Kazuto Tominaga 21

Category Theoretical Formalization of Autopoiesis from Perspective of Distinction between Organization and Structure
Tatsuya Nomura 31

Computing Substrates and Life
Soichiro Tsuda, Klaus-Peter Zauner, and Yukio-Pegio Gunji 39

The Organism 51

Effect of Multi-Level Fitnesses on the Evolution of Multicellularity in Artificial Organisms
Moritz Buck and Chrystopher L. Nehaniv 51

A Rule-Based Approach to Selecting Developmental Processes
Jan T. Kim 63

Evolutionary Robustness of Differentiation in Genetic Regulatory Networks
Johannes F. Knabe, Chrystopher L. Nehaniv, and Maria J. Schilstra 75

Multi-Agent Design to Reduce Complexity of Biological Processes Description
Nicolas Parisey, Marie Beurton-Aimar, Charles Lales, and Jean-Pierre Mazat 85

The Group 95

Interaction and Emergent Behavior in Heterogeneous Groups of Artificial Agents
Sven Magg and Rene te Boekhorst 95

Emergent Sorting Patterns and Individual Differences of Randomly Moving Ant-Like Agents
Alexander Scheidler, Daniel Merkle, and Martin Middendorf 105

Evolution 117

Explicit von Neumann-style Reproduction and Genetic Relativism in Tierra
Axel von Kamp 117

On Bio-Design of Argo-Machine
Andrew Kuznetsov, Mark Schmitz, and Kristian Müller 125

On the Evolution of Chemical Organizations
Naoki Matsumaru, Pietro Speroni di Fenizio, Florian Centler, and Peter Dittrich 135

Author Index 147

Committees

Program Committee

Andrew Adamatzky, University of the West of England Bristol
Wolfgang Banzhaf, Memorial University of Newfoundland St. John's
Clemens Beckstein, Friedrich Schiller University Jena
Mark Bedau, Reed College Portland
Stefan Bornholdt, University of Bremen
Wilfried Brauer, Technical University of Munich
Thomas Christaller, Fraunhofer Institute, Sankt Augustin
Claus Emmeche, University of Copenhagen
Horst-Michael Groß, Technical University of Ilmenau
Michael Hauhs, University of Bayreuth
Martin C. Hirsch, interActive Systems Marburg and Berlin
Jan T. Kim, University of East Anglia Norwich
Bernd-Olaf Küppers, Friedrich Schiller University Jena
Hanspeter A. Mallot, Eberhard Karls University Tübingen
Thomas Martinetz, University of Lübeck
Brian Mayoh, University of Aarhus
John McCaskill, Ruhr-University Bochum
Martin Middendorf, University of Leipzig
Christian Müller-Schloer, University of Hannover
Chrystopher Nehaniv, University of Hertfordshire Hatfield
Daniel Polani, University of Hertfordshire Hatfield
Raul Rojas, Free University Berlin
Kerstin Schill, University of Bremen
Jürgen Schmidhuber, Dalle Molle Institute of Artificial Intelligence Lugano
Frank Schweitzer, Swiss Federal Institute of Technology Zürich
Andre Skusa, University of Bielefeld
Peter F. Stadler, University of Leipzig
Jochen Triesch, Frankfurt Institute for Advanced Studies
Claus Wilke, University of Texas Austin
Klaus-Peter Zauner, University of Southampton

Organizing Committee

Stefan Artmann, Friedrich Schiller University Jena
Peter Dittrich, Friedrich Schiller University Jena

Sponsors

We wish to thank the following organizations for their contribution
to the success of the 7th German Workshop on Artificial Life:

European Office of Aerospace Research and Development (EOARD),
Air Force Office of Scientific Research, United States Air Force Research Laboratory
(www.london.af.mil)

German Research Community (DFG)

Deutsche
Forschungsgemeinschaft
DFG

We also thank the Friedrich Schiller University Jena and especially the Institute of Systematic Zoology and Evolutionary Biology with Phyletic Museum (Head: Prof. Dr. Martin S. Fischer) for hospitality and providing local facilities.

PHYLETISCHES MUSEUM JENA

Artificial? Life?
A Kind of Preface to Any Possible Exploration in the Complexity of Life-as-It-Could-Be

Stefan Artmann(1) and Peter Dittrich(2)

(1) Institute of Philosophy
Friedrich Schiller University Jena
D-07737 Jena, Germany
stefan.artmann@uni-jena.de

(2) Bio Systems Analysis Group
Jena Centre for Bioinformatics and
Department of Mathematics and Computer Science
Friedrich Schiller University Jena
D-07743 Jena, Germany
dittrich@minet.uni-jena.de

New research programmes need general, clear, and flexible ideas of the objects they study, the methods they use, and the aims they want to achieve. This is particularly true for fields of science, such as Artificial Life (AL), that are distinguished by a high level of interdisciplinary cooperation. The now classical definition of AL, given by Christopher G. Langton in the very first volume of AL proceedings, has successfully played the role of a point of reference for the large variety of contributions to research into possible life :

> "Artificial Life is the study of man-made systems that exhibit behaviors characteristic of natural living systems. It complements the traditional biological sciences concerned with the *analysis* of living organisms by attempting to *synthesize* life-like behaviors within computers and other artificial media. By extending the empirical foundation upon which biology is based *beyond* carbon-chain life that has evolved on Earth, Artificial Life can contribute to theoretical biology by locating *life-as-we-know-it* within the larger picture of *life-as-it-could-be*." [1]

The task of abstracting the essential features of living systems in order to construct artificial life-like and living systems soon aroused the curiosity of a steadily growing number of information and computer scientists, biologists and chemists, mathematicians, physicists, and philosophers. In 1995, the first German Workshop on Artificial Life (GWAL) was organised by young researchers who were interested in this newly emerging science and wanted to discuss their projects and results in an open, informal, and - despite its title - international atmosphere. The 7th GWAL, which is held in Jena 2006, continues the successful series of these workshops. It is attended by young scientists, not only from Germany, but also from other European countries and from Japan.

Jena is most certainly not a blank sheet of paper in the history of biology - just remember that Ernst Haeckel, one of the most famous evolutionary biologist and natural philosopher of the 19th century, lived in Jena for more than fifty years and founded the institutions where GWAL-7 takes place: the Zoological Institute of the Friedrich Schiller University and the first museum of evolutionary biology worldwide, the Phyletic Museum. So it might not come as a surprise that the Jena atmosphere of philosophical thinking about biology inspired us, the organisers of GWAL-7, to take the opportunity to preface these proceedings with short reflections on the future of AL. Our preliminary thoughts do not want to give final answers but are just meant to initiate a discussion that might be taken up at GWAL-8.

* *
*

Is AL engineering, science, or slogan? Nils J. Nilsson [2] proposed these three alternatives as different possibilities to characterise the nature of research in a field that is often considered AL's older sibling: Artificial Intelligence (AI). Yet ALifers who want to develop a philosophical understanding of their research can reasonably choose one of Nilsson's alternatives only after they have made a sequence of decisions on three fundamentals of AL: its methodology, its epistemology, and its ontology[1].

Nilsson [2] distinguishes a broad from a narrow view on AI. Broadly defined, AI covers every process that is involved in intelligent behaviour in human beings and animals. Analogously, 'broad AL' encompasses any process that is involved in the behaviour of living systems, so that AI becomes a part of AL. Research in broad AL is data-driven: the more empirical information ALifers have about living and life-like systems, the better they are able to construct objects that imitate organisms more and more realistically. By contrast, Nilsson's narrow view on AI delimits a core topic, the cognitive processes of reasoning and planning, and discriminates between more central and more peripheral processes involved in intelligent behaviour. Analogously, 'narrow AL' concentrates on few principal attributes of living systems that account for their behaviour best. To select these substantial qualities in a reasonable way, AL must be theory-driven: its most important task is to develop formal theories of life and to test them by finding out which of them can successfully orientate the experimental search for (*in vivo*, *in vitro*, or *in silico*) data that confirm its hypotheses about the substantial qualities of living systems.

The decision between a broad and a narrow conception of AL amounts to the methodological choice between a data- and a theory-driven research programme. Though any empirical science involves theories and data, such a distinction is sensible since it allows to recognise that in a particular research programme theories may be gradually more important than data (or vice versa) [3]. We think that AL should be theory-driven; otherwise, it runs the risk of degenerating into a kind of digital taxonomy [4]. Compared to traditional biology - a science that struggles to become more theory-driven - the need of AL for a systematically developed and experimentally tested theoretical framework is even stronger since the data of AL are not limited to the results of natural evolution on Earth.

The methodological difference between theory-driven and data-driven research programmes might be better suited to characterise AL in contradistinction to traditional biology than Langton's stricter opposition between analysis and synthesis [1]. Analysis reduces a complex object to its simpler parts and their mutual relations; synthesis puts simple elements together to generate

[1]By "epistemology" we mean the study of knowledge, and by "ontology" the study of being. These terms denote traditional philosophical disciplines. Note that the term "ontology" has a different meaning in computer science.

a complex object whose structure and function are to be explored. Yet there does exist neither pure analysis nor pure synthesis in science. A scientist has to analyse the object he tries to synthesise in order to develop an idea of what elements can generate this object. Before deciding how the analytical reduction to simpler elements will proceed, the scientist must ask which parts possibly discovered in the next analytical step could synthesise the given object. Every scientific study is analytico-synthetical, but the methodological emphasis may change from one research programme to the next. The difference between theory- and data-driven research allows to describe such a possible shift of emphasis: if a research programme is theory-driven, synthesis can become gradually more important than analysis since a good theory is the best conceptual means to guide synthesis.

If AL is a theory-driven research programme, an epistemological question seems to arise necessarily: does a theory specify objective criteria for determining the degree of life-likeness that is shown by an object generated in the framework of this theory? Yet this epistemological question might be misleading, and even dangerous, since it invites to discuss the ontological nature of AL objects instead of the epistemological scope of AL theories. Then one is inclined to ask: do the objects that are constructed by ALifers really live? The discussion about strong versus weak AL is a good example of substituting ontology for epistemology [5].

A reflection about what kind of science AL is can help evade such a slippery slope from epistemology to ontology. Even if theory-driven, AL is not a purely formal science like mathematics or logic. Yet it possesses also an epistemological status different from traditional empirical science. Like cybernetics and AI, information and system theory, game and decision theory, AL is a structural science, i.e. a transdisciplinary formalisation programme that helps discover interdisciplinary analogies between research problems coming from different sciences [6]. Any empirical object that can be described by a mathematical structure counts likewise as one of its instantiations - however dissimilar these instantiations may ontologically be. The relation between a mathematical structure and the empirical objects that instantiate it (in terms of logical semantics: the relation between a theory and its models) is much more important than the ontological relations between these objects. In theory-driven AL, ontological problems are thus to be redefined as semantical questions about the epistemological relation between theories of life and the objects these theories describe and help generate.

If AL is considered a structural science, then a traditional ontological opposition that is very important for Langton's classical definition of AL is also neutralised: the opposition between artificial and natural entities [1]. He defines 'artificial' as being man-made, and refers in this respect to Herbert A. Simon's lectures on 'The Sciences of the Artificial' [7]. Yet Simon does not want to simply identify artificial with man-made objects. For Simon, every system that is adapted to its environment is artificial: any organism that evolved by natural selection belongs to this class of entities.

Simon's discussion of the difference between natural and artificial entities indicates that this ontological opposition might not be suited for defining the objects of AL. Instead, another conceptual element of his classical definition should be considered for this task: the modal concept of possibility. AL is the study of *possible* life, life-as-it-could-be [1]. This definition should be systematically developed in the framework of a modal semantics that is based on a theory of possible worlds [8]. Entities that are possible in our real world are only particular examples of objects that are real in some possible world, and the experiments of AL are experiments on possible life in our world that are simultaneously conceivable as real experiments on real life in possible worlds - whatever ontology of artificiality and naturalness the experimenters may have [6].

Coming back to Nilsson's question whether AI (and, analogously, AL) is engineering, science, or slogan [2], our picture of AL-as-it-should-be leads us to the following answers. First, AL is not a fashionable slogan that will be outmoded in the near future if it further develops its methodological and epistemological conscience in a systematic way. Second, AL is a kind of engineering: it is modal engineering in the sense that the objects it successfully realizes remain possible entities, real specimen of possible life.The traditional empirical sciences and traditional engineering will always regard AL entities as imitations of the real stuff. To develop a systematic understanding of its complex epistemology, AL as engineering must be theory-driven. Third, if AL succeeds in developing a general theory of living systems, it becomes the structural science of possible life. Then the next decisive step AL will take might be to cut any mimetic relation between the objects it generates and real organisms so that the former entities are studied for themselves - as objects without an original in physical reality [9].

* *
*

We do not know whether the contributors to the GWAL-7 proceedings will subscribe to our general ideas of AL. The gist of our philosophical reflection is that in order to do excellent research in AL nobody needs to adopt a particular ontology of life - ALifers should just follow fundamental methodological standards of logical coherence and empirical testability, and make the semantics of their theories as clear as possible. This is surely the case for all papers that are collected in this proceedings volume. They were reviewed, and recommended as talks, by members of the GWAL-7 programme committee. We thank the contributing authors and the invited speakers, Prof. Mark A. Bedau, Prof. Jürgen Schmidhuber, and Prof. Sanjay Jain, for their high-quality papers and talks, and the referees for their fast and thorough reviewing.

The GWAL-7 proceedings are organised in topical chapters. The first chapter collects papers that discuss general concepts for the analysis and synthesis of living and life-like systems. In the second and the third chapter, the papers study two different levels of biological organisation, individual living systems and groups of such systems. The papers of the last chapter discuss evolutionary processes.

There are much more people whose support we gratefully acknowledge, particularly Prof. Martin S. Fischer, the staff of his Institute of Systematic Zoology and Evolutionary Biology and of the Phyletic Museum for their generous hospitality. The members of Peter Dittrich's Bio Systems Analysis Group were always helpful in organisational matters. Wiebke Iffert managed the GWAL-7 website in a very professional manner. During the workshop students of bioinformatics take care of coffee and cake: thanks a lot! We thank the staff of Aka and IOS press for its professional help in planning and producing this volume. Last not least, we are indebted to the sponsors of GWAL-7, the EOARD and the DFG, for their financial help which allowed us to invite speakers from abroad and to publish the proceedings.

References

[1] Langton, C.G.: Artificial Life. In: Langton, C.G. (ed.): Artificial Life. SFI Studies in the Sciences of Complexity, Proceedings Vol. VI. Addison-Wesley, Redwood City/CA (1989) 1-47

[2] Nilsson, Nils J.: Artificial Intelligence: Engineering, Science, or Slogan? In: AI Magazine 3 (1981) 2-9

[3] Langley, P.; Simon, H.A.; Bradshaw, G.L.; Zytkow, J.M.: Scientific Discovery. Computational Explorations of the Creative Processes. MIT Press, Cambridge/MA (1987)

[4] Fontana, W.; Wagner, G.; Buss, L.W.: Beyond Digital Naturalism. In: Langton, C.G. (ed.): Artificial Life. An Overview. MIT Press, Cambridge/MA (1995) 211-227

[5] Boden, M.A. (ed.): The Philosophy of Artificial Life. Oxford University Press, Oxford (1996)

[6] Artmann, S.: Artificial Life as a Structural Science. In: Philosophia naturalis 40 (2003) 183-205

[7] Simon, H.A.: The Sciences of the Artificial. MIT Press, Cambridge/MA (1969)

[8] Lewis, D.K.: On the Plurality of Worlds. Blackwell, Oxford (1986)

[9] Lem, S.: Summa technologiae. Suhrkamp, Frankfurt am Main (1981)

Automated Design of Artificial Life Forms from Scratch

Mark Bedau

ProtoLife SRL, Venice, Italy
Reed College, Portland, Oregon
European Center for Living Technology, Venice, Italy

Abstract (Invited Talk)

Artificial life has three branches, corresponding to three methods for synthesizing life-like systems. “Soft” ALife creates software systems like Tierra, “hard” ALife creates hardware like autonomous robots, and “wet” ALife creates biochemical systems in the laboratory. As part of the 6th Framework FET IST Integrated Project “Programmable Artificial Cell Evolution (PACE)”, ProtoLife and the European Center for Living Technology are developing a wet ALife evolutionary design methodology for creating complex functional chemical systems in the laboratory. Our ultimate aim is eventually to create artificial life forms from scratch. This talk will describe this project, indicate recent progress, and discuss its larger social and ethical ramifications.

The Emergence and Collapse of Complexity: Perspective from an Evolving Network Model

Sanjay Jain

Department of Physics and Astrophysics
University of Delhi
India

Abstract (Invited Talk)

Among the many puzzles relating to the origin of life, a central one is how an “organizational structure” could have emerged, characterized by a complex network of interacting molecules each playing a specific “role” in the larger organization. A similar question is relevant for a human society wherein a complex network of agents playing different roles emerges and evolves. The talk will present a mathematical model in which complex network structure self-organizes around a seed “autocatalytic set”. The ACS is a sub-network structure that has a “self-sustaining” property, and as it expands while retaining this property nodes come to acquire different roles in the organization depending upon their location in the network. Thus the model describes a mechanism by which complex organizational structure can spontaneously arise.

Complex systems in the real world also exhibit catastrophes such as mass extinctions in the biosphere, collapses of civilizations, stock market crashes, etc. Many of these catastrophes are not traceable to any external shock, and are probably due to some internal “organizational fragility” that develops naturally during the course of their evolution. The above model also exhibits sudden crashes, followed by recoveries. These will be described in terms of the changes that occur in the underlying organizational structure of the ACS. It turns out that when the system is poised for a crash, it has a characteristic network structure. Some new results will be presented showing that an increase in the diversity of its nodes enhances system robustness to crashes by providing a greater redundancy of sustenance pathways in the “core” of the organization. The model also suggests a direction towards formalizing the notion of “innovation” in complex evolving systems; most of its ups and downs can be viewed as consequences of “innovations”, defined appropriately, that occur in the course of evolution.

Goedel Machines

Jürgen Schmidhuber

Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA)
Lugano, Switzerland

Institut für Informatik
Technische Universität München, Garching, Germany

Abstract (Invited Talk)

We may use Goedel's self-reference trick to build a universal problem solver. A Goedel machine is a computer whose original software includes axioms describing the hardware and the original software (this is possible without circularity) and some formal goal in form of an arbitrary user-defined utility function (e.g., cumulative future expected reward in a sequence of optimization tasks). The original software also includes a proof searcher which uses the axioms to systematically make pairs ("proof", "program") until it finds a proof that a rewrite of the original software through "program" will increase utility. The machine can be designed such that each self-rewrite is necessarily globally optimal in the sense of the utility function, even those rewrites that destroy the proof searcher.

Information and closure in systems theory

Nils Bertschinger, Eckehard Olbrich, Nihat Ay, Jürgen Jost

Max Planck Institute for Mathematics in the Sciences
Inselstr. 22
D 04103 Leipzig, Germany

Abstract

The notion of closure plays a prominent role in systems theory where it is used to identify or define the system in distinction from its environment and to explain the autonomy of the system. Here, we present a quantitative measure, as opposed to the already existing qualitative notions, of closure.
We shall elaborate upon the observation that cognitive systems can achieve *informational closure* by modeling their environment. Formally, then, a system is informationally closed if (almost) no information flows into it from the environment.
A system that is independent from its environment trivially achieves informational closure. Simulations of coupled hidden Markov models demonstrate that informational closure can also be realized non-trivially by modeling or controlling the environment. Our analysis of systems that actively influence their environment to achieve closure then reveals interesting connections to the related notion of autonomy.
This discussion will then call into question the system-environment distinction that seems so innocent to begin with. It turns out that the notion of autonomy depends crucially on whether, not just the state observables, but also the dynamical processes are attributed to either the system or the environment. In that manner, our conceptualization of informational closure also sheds light on other, more ambitious notions of closure, e.g. organizational closure, semantic closure, closure to efficient cause or operational closure, intended as a fundamental (defining) concept of life itself.

1 Introduction

Our theoretical interest concerns the type of system that is a unity for and by itself and not only for an external observer distinguishing some entity from the rest of the world. This requires a system that can be described as a whole without reference to its environment. In systems theory, this property is usually referred to as closure.
In this regard, one encounters several notions of closure in the literature: autopoiesis as organizational closure (Maturana and Varela [1]), closure to efficient cause (Robert Rosen [2]), semantic closure (Howard Pattee [3]), or operational closure (Niklas Luhmann [4]). These concepts of closure play an important role in the architecture of systems theory, because they are used to

1. define the system (in distinction to its environment) and to
2. explain the autonomy of the system.

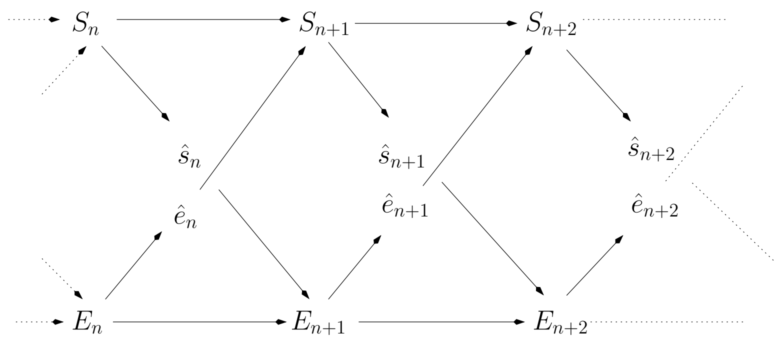


Figure 1: *The system S and the environment E interact through the channels $\hat{s}$ and $\hat{e}$. The figure shows the temporal dependencies of this interaction.*

An autopoietic system, for example, is said to be organizationally closed, because the processes that constitute the organization of the system are themselves maintaining/reproducing the conditions for their own existence. Thereby, the system also defines its own boundary that separates itself from its environment. This self-referential distinction from its environment therefore gives rise to the specific autonomy of such a system. Consequently, in systems theory, closure properties and autonomy are considered to be closely related concepts which are both at the heart of defining the system itself.

These aforementioned attempts, however, either remain vague (Luhmann, Maturana), lead to concerns about their formal consistency ([5]), or are too abstract to work with ([2]). Therefore, the formalization of closure remains a central issue in the formalization of systems theory.

As a starting point, we use a closure type phenomenon that can be observed in cognitive systems. Such systems are assumed to be capable of reducing the information flow from the environment into the system by modeling the environment. We shall call this “informational closure”. We think that this concept can also contribute an abstract notion of “modeling” that does not depend upon the identification of certain structures in the system as explicit models or representations.

Informational closure is, arguably, not enough to completely capture all properties of closure, in particular the ones required to derive the existence of the system itself. It is, however, more amenable to quantification since it should not be considered to be an all-or-nothing phenomenon. A system can only be informationally closed with respect to those features of the environment that can be modeled. Unpredictable events that nevertheless do affect the system clearly have to give rise to an information flow into the system.

In this paper, after introducing the basic setup and the notation, we propose a measure for informational closure. In section III we apply our measure to simple hidden Markov models. There, we focus on non-trivial closure, i.e. systems that feature low information flow from the environment, but nevertheless contain mutual information about their environment. We also distinguish the case of passive systems, which cannot influence the environment, from the case of active ones that are able to change the environment. This not only allows us to propose a measure of the influence a system can exert upon its environment, called self-efficacy, but also leads to a discussion of the related notion of autonomy. We conclude with a summary of our information theoretic description of closure in the context of systems theory and provide an outlook on future research directions.

2 Informational Closure

The basic setting we want to study is shown in Fig. (1). We assume that we have observables S_n describing the state of the system and observables E_n describing the state of the environment at time t_n. At the beginning, these are chosen according to the system-environment distinction of an external observer. According to our view, this distinction is justified insofar as the system, defined by that distinction, can really achieve closure.

The notion of informational closure refers to a situation where the information flow between the environment and the system tends to zero. We measure the information flow from the environment into the system $J_n(E \to S)$ by the conditional mutual information [1]:

$$\begin{aligned} J_n(E \to S) &= MI(S_{n+1}; E_n | S_n) && (1) \\ &= H(S_{n+1}|S_n) - H(S_{n+1}|S_n, E_n) && (2) \\ &= H(E_n|S_n) - H(E_n|S_n, S_{n+1}) \ . && (3) \end{aligned}$$

Here, $H(A)$ denotes the entropy of the probability distribution of the random variable A. For the sake of simplicity, we restrict ourselves to discrete observables, i.e.

$$H(A) = -\sum_{i=1}^{N} p(a_i) \log_2 p(a_i) \tag{4}$$

where the random variable A takes the value a_i with probability $p(a_i)$. For two random variables A and B, $H(A, B)$ denotes the entropy of the joint probability distribution $p(A, B)$ and

$$H(A|B) = H(A, B) - H(B) \tag{5}$$

the conditional entropy of A given B. The mutual information between two random variables A and B is defined as

$$MI(A; B) = H(A) - H(A|B) \ , \tag{6}$$

and the conditional mutual information between A and B given C as

$$MI(A; B|C) = H(A|C) - H(A|BC) \ . \tag{7}$$

The latter measures the reduction of the uncertainty of A given B if C is known additionally, which can be interpreted as the "information flow" from A to B (or in the opposite direction) not transmitted via C.
For the information flow from the environment into the system, as defined in Eq. 1, the following useful identity holds:

$$MI(S_{n+1}; E_n|S_n) = MI(S_{n+1}; E_n) - (MI(S_{n+1}; S_n) - MI(S_{n+1}; S_n|E_n)) \ . \tag{8}$$

This expresses the information flow as the difference between

- the information that S_{n+1} contains about the environment, $MI(S_{n+1}; E_n)$
- and the mutual information between consecutive system states that is related to the environment, $MI(S_{n+1}; S_n) - MI(S_{n+1}; S_n|E_n)$.

We shall now discuss different ways of how a system can achieve informational closure. First there is a trivial case:

[1] This is also known as "transfer entropy" [6] and can be considered as a quantitative expression for the Granger causality [7].

A) Independence: The system and the environment are independent stochastic processes. This implies in particular

$$\begin{aligned} MI(S_{n+1};E_n) &= 0 \qquad \text{and} \\ MI(S_{n+1};S_n) - MI(S_{n+1};S_n|E_n) &= 0 \end{aligned}$$

and therefore informational closure (8).

The "informational closure" becomes non-trivial if the state contains information about the environment, i.e. $MI(S_{n+1};E_n) \neq 0$:

B) Adaptation: The state of the system contains full information about the part of the environment that interacts with the system, i.e. $H(\hat{e}_n|S_n) = 0$. By the independence structure as stated in Fig. (1) and Eq. (3), this implies closure $J_n = 0$.

For this case, however, two subcases should be distinguished:

B1) Passive adaptation: The system is driven by the environment and adapts passively to all changes in the environment.
In the case of an environment, that appears deterministic to the system, $H(\hat{e}_{n+1}|\hat{e}_n) = 0$, this can be achieved by simply copying the observation of the environment into the system, $S_{n+1} = \hat{e}_n$.

B2) Modeling: The system reaches synchronization and internalizes the correlations observed in the environment by building up own structures.

If the system can act in the environment there is an additional possibility to achieve closure:

C) Control: The system tries to maximize the information flow between its actions and its sensory inputs (called *empowerment* in [8]), i.e. it produces changes in the environment such that these in turn change the state of the system most efficiently. If the actions $\hat{s}$ of the system are functions of the system state, then the consecutive state of the system can anticipate the effect on the environment, which also leads to informational closure.

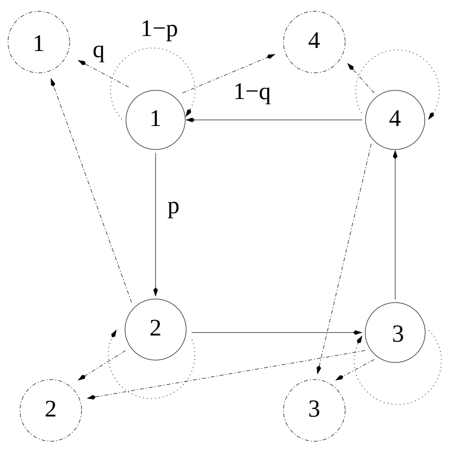
Since, in the following, we are mainly interested in the case of non-trivial closure we define

$$\begin{aligned} NTIC_m &:= MI(S_{n+1};E_n,\ldots,E_{n-m}) - MI(S_{n+1};E_n,\ldots,E_{n-m}|S_n) \\ &= MI(S_{n+1};E_n,\ldots,E_{n-m}) - MI(S_{n+1};E_n|S_n) \qquad (9) \end{aligned}$$

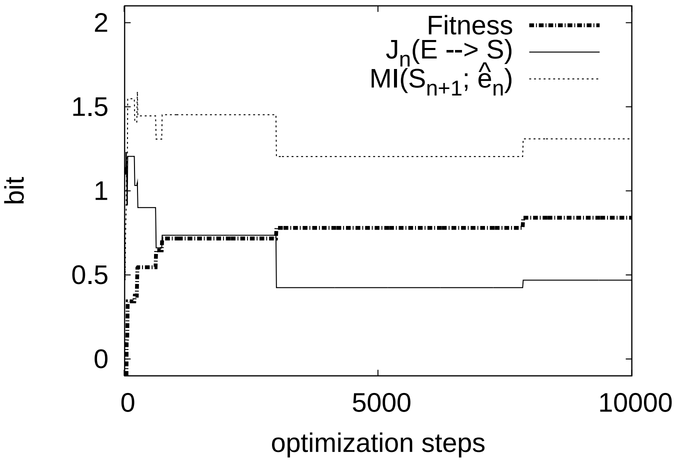
as a measure for the amount of non-trivial informational closure. Note that a large value of this measure does not ensure a low information flow. It just requires that the system contains more information about the environment than it gathered at this time-step. It should therefore not be considered as a replacement for closure, i.e. low information flow, but as a complementary measure that quantifies the amount of non-trivial closure when there is closure.

3 Simulations

As a simple example we consider an environment described by the hidden Markov model shown in Fig. 2A. The environment can either be deterministic ($p = 1$) or non-deterministic ($p = 0.9$).



A



B

Figure 2: *A) Hidden Markov model used as environment. B) Optimization of Closure and Mutual Information by simulated annealing.*

In both cases, the state observations $\hat{e}$ are noisy ($q = 0.8$), which makes the visible dynamics quite unpredictable ($H(\hat{e}_{n+1}|\hat{e}_n) = 1.23$ bits in the case $p = 1$). This excludes passive adaptation B1 as a strategy to achieve non-trivial informational closure.
The non-deterministic environment is then used to drive a deterministic system, i.e. $S_{n+1} = F(S_n, \hat{e}_n)$. We consider systems with two and four internal states. Note that four internal states are required to fully model the hidden environmental structure. First, we do not consider the case where the system is acting upon the environment, which makes option C (control) unavailable.
In our simulations (see Fig. 2B), we optimize the transition structure F via simulated annealing using the fitness function

$$MI(S_{n+1}; \hat{e}_n) - MI(S_{n+1}; \hat{e}_n|S_n) , \tag{10}$$

which enforces non-trivial closure through the system gaining mutual information with the environment. Note that in contrast to $NTIC_m$ for $m = 0$ as defined above, we use $\hat{e}_n$ instead of E_n, which makes this measure available to the system itself since it only depends on quantities that either belong to or are observable by the system. The original version for $NTIC_m$ is considered to be more suitable for an external observer since it fully exploits the independence structure of the system-environment interaction.

As expected, due to the non-determinism of the environment, full closure cannot be achieved, but the best system with four internal states that was found after 10000 optimization steps is able to model the hidden environmental dynamics, i.e. $MI(S_n; E_n) > MI(S_n; \hat{e}_n)$. This system achieves full closure, $J_n = 0$, and maximal mutual information (2 bit) when coupled to the deterministic environment ($p = 1$), which shows that it has extracted the deterministic part of the system dynamics (Fig. 3 A).
An optimized system with two internal states is not able to adequately represent the environmental dynamics as seen in Fig. 3 B. Even when coupled to the deterministic environment, the system is far from being informationally closed and contains only little information about the environment.

To investigate option C), where the system can control the environment, we change the environment such that:

1. it rotates stochastically, as before, if the system emits a "null" action $\hat{s} = n$
2. and is reset to state 1 if the system emits a reset action $\hat{s} = r$.

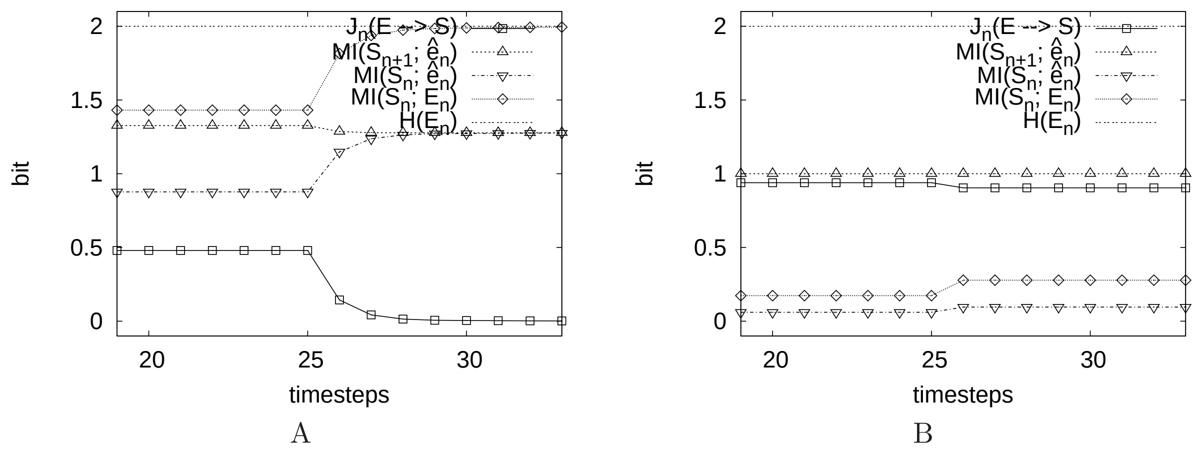


Figure 3: *A) Reaction of the system with four internal states when coupled to the deterministic version of the environment (p is changed from 0.9 to 1 after 25 time-steps). B) Same as A), but for the system with two internal states.*

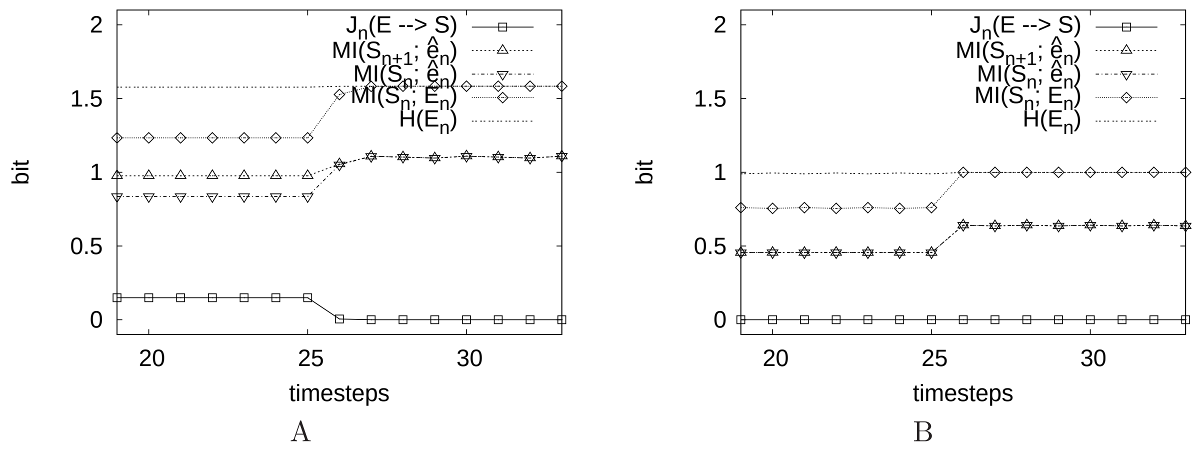


Figure 4: *A) Reaction of the active system with four internal states when coupled to the deterministic version of the resettable environment (p is changed from 0.9 to 1 after 25 time-steps). B) Same as A), but for the system with two internal states.*

As before, we optimize systems with two and four internal states to achieve non-trivial closure. But now, not just the transition structure of the system, but also the actions emitted by the system, were optimized. As can be seen from Fig. 4, this allows for a much better informational closure as compared to the corresponding passive system (Fig. 3). This difference is especially large in the case of two internal states. By using actions to restrict the environmental dynamics, even the two state system can achieve full informational closure and model the deterministic part of the restricted environment (Fig. 4 B). The system with four internal states was found to restrict the system to three states instead of modeling the uncontrolled four state rotation. To quantify the control a system exerts on its environment, we introduce the following measure for *self-efficacy*:

$$\begin{aligned} E_s \quad &:= \quad MI(S_{n+1}; S_{n-1}|E_{n-1}) \\ &\quad\;\; - MI(S_{n+1}; S_{n-1}|S_n, E_{n-1}) - MI(S_{n+1}; S_{n-1}|E_n, E_{n-1}) \end{aligned} \tag{11}$$

Inspired by the *empowerment* measure in [8], this quantity measures the influence of the system state S_{n-1} on S_{n+1} (a generalization to longer time spans can be defined along the same lines) that is transmitted through the environment and simultaneously mediated by the system.

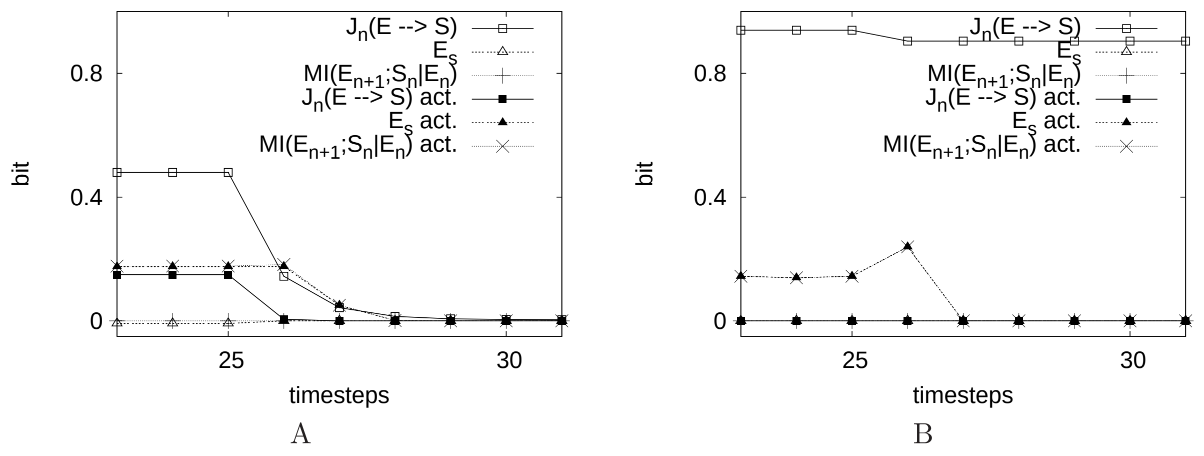


Figure 5: *A) Self-efficacy and information for the systems (passive and active) with four internal states when coupled to the deterministic version of the resettable environment (p is changed from 0.9 to 1 at time-step 25). This figure shows the adaptation to a new environment as in figures 3 and 4 A, but in more detail and compares the active and passive system also with respect to their self-efficacy. B) Same as A), but for the system with two internal states. (Note: E_s is zero in the passive case and therefore not visible in the plot.)*

In contrast to [8], were empowerment is defined as the channel capacity from actions to future sensor inputs, we do not assume a "free will" that can freely choose actions to optimize the transmitted information. Instead, we only consider actions that can occur for the observed internal states of the system and therefore base our measure on the information that actually is transmitted from S_{n-1} to S_{n+1}.
To account for the fact that the information has to be transmitted through the environment, we also subtract the part of the mutual information that corresponds to correlations only inside the system, $MI(S_{n-1};S_{n+1}|E_{n-1})$. We also require that the influence the system can observe at time-step $n+1$ is still known to the system, i.e. internally represented. To take this into account, we subtract the part of the mutual information that is only transmitted through the environment, i.e. $MI(S_{n-1};S_{n+1}|S_n)$ which can be seen as a measure of "self-surprise".
The conditioning on E_{n-1} of all quantities removes any information that is shared between E and S already at time-step $n-1$ and can therefore not be attributed to the effect the system had on the environment from time-step $n-1$ to time-step n.
Considering the Markovian structure of the system-environment interaction, i.e. $MI(S_{n-1};S_{n+1} \mid S_n, E_n) = 0$, Eq. (11) can be considered as a conditional version of the co-information (compare [9]) shared between S_{n-1}, S_n, S_{n+1} and E_n (as shown in Appendix A). Like the co-information, this quantity can become negative. We do not consider this as a problem in this case, since it fulfills the intuitive requirement

$$E_s \leq MI(E_n;S_{n-1}|E_{n-1}). \tag{12}$$

This shows that the self-efficacy of the system is bounded from above by the information flow from the system into the environment.

In Fig. 5, we show the self-efficacy (11) for the systems discussed above. As expected, in the case of purely observing systems that do not influence the environment, the information flow from the system into the environment vanishes and the self-efficacy is slightly smaller than zero. The active systems show a positive self-efficacy that is utilized to achieve better closure when coupled to the stochastic version of the environment. They react to the deterministic environment by a short (slight) increase of the self-efficacy to quickly achieve full closure. Note that the self-efficacy drops to zero when the system is fully synchronized to the environment.

This is due to the fact that in this case no information flow into the environment can be observed, similar to the case of a system that achieves closure by copying the environment as discussed above. Here, the environment can be considered to copy the deterministic dynamics of the acting system and is therefore closed with respect to the system, i.e. the influence of the system is impossible to detect since the environment never uses more states than enforced by the system.

4 Closure and Autonomy

The problem of correct attribution of control also arises in the related notion of autonomy. As described in detail in [10] under the assumption that the system cannot control the environment, corresponding to the cases A and B discussed above, a suitable autonomy measure is given by

$$\begin{aligned} A_m &= MI(S_{n+1}; S_n | E_n, \ldots, E_{n-m}) \qquad (13) \\ &= H(S_{n+1}|E_n, \ldots, E_{n-m}) - H(S_{n+1}|S_n, E_n, \ldots, E_{n-m}) \\ &= H(S_{n+1}|E_n, \ldots, E_{n-m}) - H(S_{n+1}|S_n, E_n) \end{aligned}$$

This describes autonomy as the difference between non-heteronomy, measured by $H(S_{n+1}|E_n, \ldots, E_{n-m})$ as the extent to which the system state cannot be determined from the environment, and the intrinsic randomness that also the system does not control ($H(S_{n+1}|S_n, E_n)$).
If instead all mutual information shared between system and environment is attributed to the system, i.e. the system is assumed to be in maximal control of its environment, then the following autonomy measure is more adequate:

$$A^* = MI(S_{n+1}; S_n)$$

Here, the mutual information between successive system states just reflects the autonomy of the system.
These two autonomy measures are closely related to the non-trivial informational closure:

$$NTIC_m = A^* - A_m \qquad (14)$$

In the case of trivial informational closure, e.g. if the system is independent of the environment, the two autonomy measures agree. When a system achieves non-trivial informational closure, for example by modeling the environment, we can observe the following

- A^* also has to be positive, actually bigger than $NTIC_m$, since all the environmental correlations that the system can model have to be reflected within the system state.
- A_m can be quite small, because a system that accurately models its environment can be predicted quite well from observation of the environment. An external observer would therefore not attribute much autonomy to such a system since he/she is able predict its state.

Rewriting Eq. (14), we can relate autonomy directly to informational closure as measured by the information flow $J_n(E \to S)$:

$$\begin{aligned} A^* &= A + NTIC \\ &= A + MI(S_{n+1}; E_n, \ldots, E_{n-m}) - J_n(E \to S) \\ \Rightarrow \quad J_n(E \to S) &= A^* - A - MI(S_{n+1}; E_n, \ldots, E_{n-m}) \end{aligned}$$

This demonstrates that a system exhibiting certain internal regularities as measured by $A^* = MI(S_{n+1}; S_n)$ can achieve informational closure either by gaining information about the environment or by increased autonomy, i.e. by becoming unpredictable or uncontrollable from the

environment.
Therefore, information about the environment, i.e. modeling, and autonomy can be considered as complementary strategies for achieving informational closure.

5 Discussion

Our concept of informational closure is a quantitative one. The way we introduce it, it depends on a system-environment distinction that is attributed to an external observer, but not specified further. Such a distinction might seem entirely arbitrary, but this arbitrariness can be reduced through the criterion of informational closure of the system. In other words we can, conversely, use our measure of informational closure to evaluate the employed distinction between system and environment. A higher informational closure then indicates a better identification of the system.
The closure concepts in the literature that we referred to in the introduction aim at a qualitative notion of closure. In particular, autopoiesis as organizational closure in the physico-chemical domain should yield a clear-cut distinction between life and death for organisms instead of a gradual difference. From our point of view, however, in a non-deterministic environment, full non-trivial closure cannot be achieved, since it would require mutual information about the environment and at the same time the vanishing of the information flow J_n. Ultimately, this leads us to the issue of operational closure versus thermodynamic openness in systems theory. Since our measure employs information-theoretic quantities defined in terms of entropies and because the environment always has a higher entropy than any system situated in it, we should naturally expect, that a system, which needs to act adaptively, has to get some new information from the environment.
Also, the relationship of informational closure with autonomy, as developed here, seems relevant in this regard. Our analysis shows that a crucial point for the definition of autonomy is the attribution of control to the system or to the environment. This again brings us back to the system-environment distinction, since we need to decide not only for the states, but also for the processes operating on these states whether they belong to the system or to the environment. Closure then reflects how adequately the system can be described in terms of its own observables. We have demonstrated that our measure for informational closure gives meaningful results, at least in simple examples where it is unambiguous what the observables are and the control flow is known, e.g. the system cannot influence the environment. Therefore, we expect our closure measure to become a valuable tool for the analysis of system-environment interactions in artificial life simulations. In other cases where the control flow is not unambiguously known, we could also derive interesting conclusions that, admittedly, are open to interpretation.
In order to further investigate the issues raised above, we will develop the concept in the following directions:

- A general framework for quantifying adaptivity, closure and autonomy based on information theory.
- Including the optimization dynamics in the description of the system and describing not only the result, but also the process of adaptation by means of information theory. As long as the system is learning it should then not be informationally closed.
- Formalization of the notion of self-observation – a further internal differentiation of the system is then needed to additionally include a self-description of the system.
- Application of the measures to autonomous robots within a closed sensory-motor loop.

References

[1] H. R. Maturana, F. J. Varela, Autopoiesis and Cognition. The Realization of the Living, Reidel, Dordrecht, 1980.

[2] R. Rosen, Life itself: A comprehensive enquiry into the nature, origin and fabrication of life, Columbia University Press, New York, 1991.

[3] H. H. Pattee, Evolving self-reference: matter, symbols, and semantic closure, Communication and Cognition – Artificial Intelligence 12 (1-2) (1995) 9–28.

[4] N. Luhmann, Probleme mit operativer Schließung, in: Soziologische Aufklärung 6, Westdeutscher Verlag, Opladen, 1995, pp. 12–24.

[5] P. Bourgine, F. J. Varela, Towards a practice of autonomous systems, in: F. J. Varela, P. Bourgine (Eds.), Toward a Practice of Autonomous Systems, MIT Press, Cambridge, Massachusetts, 1991.

[6] T. Schreiber, Measuring information transfer, Phys. Rev. Lett. 85 (2000) 461.

[7] C. W. J. Granger, Investigating causal relations by econometric models and cross-spectral methods, Econometrica 37 (1969) 424–438.

[8] A. S. Klyubin, D. Polani, C. L. Nehaniv, Empowerment: An universal agent-centric measure of control, in: 2005 IEEE Congress on Evolutionary Computation, 2005, http://homepages.feis.herts.ac.uk/ comqdp1/publications.html.

[9] A. J. Bell, The co-information lattice, in: S. Amari, A. Cichocki, S. Makino, N. Murata (Eds.), Proceedings of the Fifth International Workshop on Independent Component Analysis and Blind Signal Separation: ICA 2003, 2003.

[10] N. Bertschinger, E. Olbrich, N. Ay, J. Jost, Autonomy: an information theoretic perspective, for the Workshop on autonomy, ALifeX conference, to appear (2006).

Appendix A

In terms of entropies, Eq. (11) can be rewritten as follows

$$\begin{aligned}
& MI(S_{n+1};S_{n-1}|E_{n-1}) - MI(S_{n+1};S_{n-1}|S_n,E_{n-1}) - MI(S_{n+1};S_{n-1}|E_n,E_{n-1}) \\
= \; & MI(S_{n+1};S_{n-1}|E_{n-1}) - MI(S_{n+1};S_{n-1}|S_n,E_{n-1}) - MI(S_{n+1};S_{n-1}|E_n,E_{n-1}) \\
& +MI(S_{n+1};S_{n-1}|S_n,E_n,E_{n-1}) \\
= \; & H(S_{n+1}|E_{n-1}) + H(S_{n-1}|E_{n-1}) - H(S_{n+1},S_{n-1}|E_{n-1}) \\
& - [H(S_{n+1},S_n|E_{n-1}) - H(S_n|E_{n-1}) + H(S_{n-1},S_n|E_{n-1}) - H(S_n|E_{n-1}) \\
& -H(S_{n+1},S_n,S_{n-1}|E_{n-1}) + H(S_n|E_{n-1})] \\
& - [H(S_{n+1},E_n|E_{n-1}) - H(E_n|E_{n-1}) + H(S_{n-1},E_n|E_{n-1}) - H(E_n|E_{n-1}) \\
& -H(S_{n+1},E_n,S_{n-1}|E_{n-1}) + H(E_n|E_{n-1})] \\
& +H(S_{n+1},S_n,E_n|E_{n-1}) - H(S_n,E_n|E_{n-1}) \\
& +H(S_{n-1},S_n,E_n|E_{n-1}) - H(S_n,E_n|E_{n-1}) \\
& -H(S_{n+1},S_n,E_n,S_{n-1}|E_{n-1}) + H(S_n,E_n|E_{n-1}) \\
= \; & H(S_{n+1}|E_{n-1}) + H(S_{n-1}|E_{n-1}) + H(S_n|E_{n-1}) + H(E_n|E_{n-1}) \\
& -H(S_{n+1},S_{n-1}|E_{n-1}) - H(S_{n+1},S_n|E_{n-1}) - H(S_{n+1},E_n|E_{n-1}) \\
& -H(S_n,E_n|E_{n-1}) - H(S_{n-1},S_n|E_{n-1}) - H(S_{n-1},E_n|E_{n-1}) \\
& +H(S_{n+1},S_n,S_{n-1}|E_{n-1}) + H(S_{n+1},E_n,S_{n-1}|E_{n-1}) \\
& +H(S_{n+1},S_n,E_n|E_{n-1}) + H(S_n,E_n,S_{n-1}|E_{n-1}) \\
& -H(S_{n+1},S_n,E_n,S_{n-1}|E_{n-1})
\end{aligned}$$

which is the co-information between S_{n-1}, S_n, E_n and S_{n+1} (conditioned on E_{n-1}).

Appendix B

Proof of Eq. (12):

$$\begin{aligned}
E_s &= MI(S_{n+1};S_{n-1}|E_{n-1}) - MI(S_{n+1};S_{n-1}|S_n,E_{n-1}) - MI(S_{n+1};S_{n-1}|E_n,E_{n-1}) \\
&= MI(S_{n+1};S_{n-1}|E_{n-1}) - MI(S_{n+1};S_{n-1}|S_n,E_{n-1}) \\
&\quad - (MI(S_{n+1};S_{n-1}|E_{n-1}) + MI(S_{n-1};E_n|S_{n+1},E_{n-1}) - MI(S_{n-1};E_n|E_{n-1})) \\
&= MI(S_{n-1};E_n|E_{n-1}) - MI(S_{n-1};E_n|S_{n+1},E_{n-1}) - MI(S_{n+1};S_{n-1}|S_n,E_{n-1}) \\
&\leq MI(S_{n-1};E_n|E_{n-1})
\end{aligned}$$

An Approach to Algorithmic Music Composition with an Artificial Chemistry

Tomoya Miura and Kazuto Tominaga

Tokyo University of Technology
Hachioji, Tokyo 192-0982 Japan
tomoyan@mikasa.tomilab.net tomi@acm.org

Abstract

The relationship between an artificial chemistry and its execution by a simulator is similar to that between a nondeterministic algorithm and one of its possible computation. In this paper, we show a possibility to use an artificial chemistry as a platform for describing and executing nondeterministic algorithms through implementing a simple algorithm for music composition as a system of our artificial chemistry. The system is obtained from the nondeterministic algorithm in a straightforward manner due to the inherent nondeterminism of artificial chemistry, and the implemented system successfully generated musical phrases. The result suggests that an artificial chemistry may be a useful platform for describing and processing nondeterministic algorithms.

1 Introduction

In the field of artificial life, the study of artificial chemistries is becoming more and more active these days. Artificial chemistries are abstract models of chemical reactions, and various kinds of artificial chemistries have been proposed [4]. Artificial chemistries are chiefly applied (1) to construct systems that show behaviour similar to that of living systems [4], or (2) to model real biochemical reactions (such studies include [6, 9, 10]). Though the aims are different, the two kinds of applications both stay within the same framework: (real or artificial) living systems and their models.

Each artificial chemistry has its reactor algorithm, with which a simulator of the artificial chemistry simulates molecular reaction in the reactor. One of the main classes of such algorithms is stochastic molecular collision; a simulator executes a system, which is described in terms of the artificial chemistry, by selecting molecules randomly and having them collide to react. This relationship between the system and the execution is similar to that between a nondeterministic algorithm and its possible computation: selecting molecules corresponds to making a choice at a choice point in the nondeterministic algorithm.

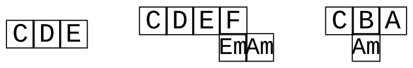
In this paper, being away from living systems for a while, we explore a possibility for an artificial chemistry to be used as a platform for nondeterministic computation. A sample problem is automatic music composition. A reason for choosing this problem is that it seems appropriate to regard music composition as a nondeterministic algorithm since there are many choice points in composing music such as selecting notes or chords. Another reason is that the problem itself is challenging since it is in the area of art.

The rest of this paper is organised as follows. After briefly explaining our artificial chemistry, we give a simple nondeterministic algorithm for composing music. Then it is implemented as a system of the artificial chemistry, and musical phrases generated by a simulator are shown. Finally, we evaluate this approach and compare with some related works.

2 Our Artificial Chemistry

This section briefly explains a simple artificial chemistry based on pattern matching and recombination [7], with which we have presented several applications [6, 8, 9, 10]. The main conceptual components of the artificial chemistry are *elements*, *objects*, *patterns*, *recombination rules* and a *system*. The present paper omits "sources" and "drains," which are also parts of the artificial chemistry, since they are not used in this study.

An *element* is a primary entity in this artificial chemistry, and it corresponds to an atom or a group of atoms in nature. It is denoted by an upper-case character or a capitalised sequence of alphanumeric characters. For example, `C`, `D`, `Am` and `Gaug7` are elements. An *object*, which corresponds to a molecule or a compound of molecules, is a character string or a stack of strings, such as those depicted below.



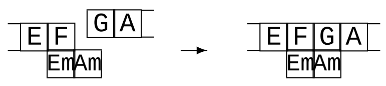
These objects are denoted by the string notations `0#CDE/`, `0#CDEF/3#EmAm/`, `0#CBA/1#Am/` respectively; the numbers represent the displacements of lines (in numbers of elements) relative to the first line. A displacement can be positive, zero or negative.

A *pattern* matches (or does not match) an object, and it may include two kinds of *wildcards*. An *element wildcard* matches any element, and it is denoted by a number surrounded by angle brackets such as `<1>`. A *sequence wildcard* matches any sequence of zero or more elements, and is denoted by a number and an asterisk with angle brackets such as `<2*>` and `<*2>`; the position of an asterisk represents the direction in which the sequence can extend.

A pattern is denoted in a similar way to an object. For example, the pattern `0#C<1>E/` matches the object `0#CDE/`; the pattern `0#<*1>EF/1#EmAm/` matches the object `0#CDEF/3#EmAm/`. Note that the displacements are meaningful and that the length of a sequence wildcard is treated as zero in the pattern notation.

A *recombination rule* transforms a collection of objects into a collection of objects, conserving elements just like a chemical reaction does. It is expressed in a manner similar to chemical formulae in terms of patterns. Following are an example rule and the illustration of recombination by the rule.

$$\texttt{0\#<*1>EF/1\#EmAm/} + \texttt{0\#GA<2*>/} \rightarrow \texttt{0\#<*1>EFGA<2*>/1\#EmAm/}$$



If this rule is applied to the objects `0#CDEF/3#EmAm/` and `0#GAB/`, the object `0#CDEFGAB/3#EmAm/` is produced and the reactants disappear. We call the left-hand side of a recombination rule *lhs*, and the right-hand side *rhs*. Each wildcard can appear only once in lhs, and it must appear once and only once in rhs.

A *system* is a construct $\langle \Sigma, R, P_0 \rangle$ where Σ is a set of elements, R is a set of recombination rules, and P_0 is the *initial working multiset*, which specifies objects in the working multiset at the initial state. The system is interpreted nondeterministically as follows: (1) Initialise the working multiset to be P_0; (2) Apply one recombination rule to a collection of objects; (3) Go to Step 2.

3 A Simple Algorithm for Music Composition

In this section, we give a simple procedure for composing music. It is somewhat based on a general composition theory for classical music as well as on one for jazz music. We consider the diatonic scale of C Major in an octave, namely, C4, D4, E4, F4, G4, A4, B4 and C5, and only the time value of eighth. The chords used are the triads on the pitches C, D, E, F, G, A and B in the C Major scale, which we will denote by C, Dm, Em, F, G, Am and Bm°, respectively.

We will generate musical phrases using the following procedure.

1. Make sequences of notes, each of which comprises eight eighth notes as one-bar melody.
2. For each bar, choose a chord that harmonises well with the first note of the bar.
3. Replace an "avoid note" in a bar (if any) with a random note.
4. Concatenate bars according to "cadence rules" to make a sequence of bars.
5. Obtain four-bar phrases from the sequence of bars.

We will explain avoid notes and cadence rules later in this section.

This procedure is a nondeterministic algorithm because each step has multiple choices of notes, chords or bars. Moreover, steps can be (or should be) executed in an interleaving manner: for example, Step 1 must be executed at least twice before Step 4 is executed because Step 4 requires two or more bars. In the following sections, each step of the algorithm is illustrated.

3.1 Making a Sequence of Notes

This step first chooses a random note as the first note of the sequence to make. Then, in order to make a smooth melody, further notes are selected so that the sequence becomes a conjunct melody, i.e., a note just above or below the previous note is chosen. For example, if E4 is chosen as the first note, the next note is either D4 or F4. If F4 is chosen at this time, the next note is E4 or G4. This process continues to make a one-bar (eight eighth note) melody.

3.2 Choosing a Chord

In this step, a chord is chosen for a one-bar melody according to the first note: a chord that includes the first note as its components is chosen. For example, if the first note of the sequence is E4, a chord is chosen for this sequence from among C, Em and Am.

3.3 Replacing Avoid Notes

A chord has a particular pitch that is dissonant with the chord. We define this pitch as an *avoid pitch* of the chord. For example, the pitch C4 does not harmonise well with the chord G, so the pitch C4 is defined as an avoid pitch of G. The list of chord and its avoid pitch is as follows: C/F4, Dm/B4, Em/F4, G/C, Am/F4 and Bm°/C (where C means both C4 and C5); the chord F has no avoid pitch.

Since the previous step chooses a chord for a bar according only to the first note of the bar, the melody may contain a note with an avoid pitch (it is generally called *avoid note*) of the chord. In order to reduce dissonance, this step replaces an avoid note with a random note. Though it is possible that this step gives another avoid note such as C5 replacing C4 for the chord G, applying this step again will replace it with another note, which is possibly not an avoid note.

3.4 Concatenating Bars

The chords are categorised into the following four functional groups: tonic (which we will denote by **T**), subdominant (**S**), dominant (**D**) and second dominant (**D**$_2$). In C Major, the tonic chords are C, Em and Am; the subdominant chords are Dm, F and Am; the dominant chords are Em, G and Bm°; and the second dominant chords are Dm and F. The following progressions of functions are typical cadences: **T-D-T**, **T-S-T** and **T-D**$_2$**-D-T**. We call these the cadence rules, and will use them to combine bars.

Each bar has been given a chord, and bars are concatenated one by one according to the cadence rules. Let this step decide which bar to add as the next bar based on the chord of the previous bar as follows.

1. Add **T** after **D** or **D**$_2$.
2. Add **S** after **T**.
3. Add **D** after **T** or **D**$_2$.
4. Add **D**$_2$ after **T**.

For example, if the first bar has the chord C, whose function is **T**, a chord of any function except for **T** can be added (because 2, 3 and 4 can be applied). If G is chosen here (which is **D**), any tonic chord (C, Em or Am) can come next (only 1 can be applied).

3.5 Obtain Phrases from a Sequence of Bars

Four-bar phrases are obtained from the sequence of bars generated in the previous step. In order for a phrase to sound naturally, we impose a restriction on phrases: one should start with a tonic chord. To achieve this, the first bar with a tonic chord is sought in the sequence, and the sequence of four bars starting at the tonic bar is obtained as one phrase. This process is repeated to generate more phrases.

4 Description of the Algorithm

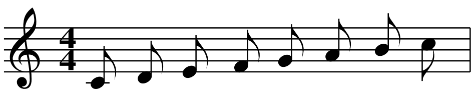
We give the description of a system that implements the algorithm. The description comprises 24 kinds of elements, 43 kinds of initial objects (1826 objects in total), and 65 recombination rules.

4.1 Design of Elements and Objects

We use the following elements.

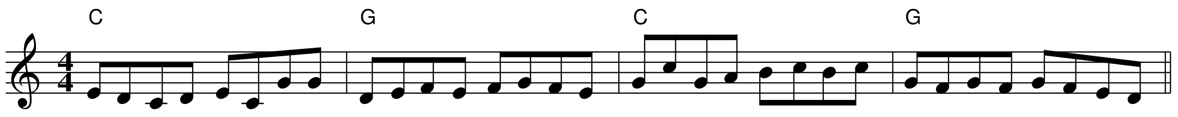
- Notes: `C4`, `D4`, `E4`, `F4`, `G4`, `A4`, `B4` and `C5`; each represents an eighth note of the pitch.
- Chords: `C`, `Dm`, `Em`, `F`, `G`, `Am` and `Bm`.
- Chord functions: `T` (tonic), `S` (subdominant), `D` (dominant) and `D2` (second dominant).
- Miscellaneous elements for control: `Degree`, `Chord`, `Avoid`, `Start` and `Stop`.

A sequence of notes is represented by a line of note elements. For example, `0#C4D4E4F4G4A4B4C5/` represents a phrase shown below.



```
0#StartE4D4C4D4E4C4G4G4D4E4F4E4F4G4F4E4G4C5G4A4B4C5B4C5G4F4G4F4G4F4E4D4Stop
/1#TCCCCCCCDGGGGGGGTCCCCCCCDGGGGGGG/
```

(a) An example object representing a four-bar phrase



(b) The music represented by the above object

Figure 1: An example object representing four-bar phrase and its music.

A bar with a chord is represented by a two-line object like `0#C4D4E4F4G4A4B4C5/0#TCCCCCCC/`, which is the same bar as above accompanied by the chord **C**. The element `T` in the second line means that the function of the chord is **T** (tonic).

A complete four-bar phrase is represented as an object that comprises four objects of the above form combined left to right, terminated by the elements `Start` and `Stop`, such as the one shown in Figure 1(a) (folded to fit in the page). This object represents the phrase shown in Figure 1(b).

4.2 Initial Objects

In the working multiset, we give the following initial objects. (The number of objects is shown in brackets.)

- Objects expressing conjunct motion [100 for each]:

  `0#Degree/0#C4D4/`, `0#Degree/0#D4C4/`, `0#Degree/0#D4E4/`, `0#Degree/0#E4D4/`, `0#Degree/0#E4F4/`, `0#Degree/0#F4E4/`, `0#Degree/0#F4G4/`, `0#Degree/0#G4F4/`, `0#Degree/0#G4A4/`, `0#Degree/0#A4G4/`, `0#Degree/0#A4B4/`, `0#Degree/0#B4A4/`, `0#Degree/0#B4C5/`, `0#Degree/0#C5B4/`.

  For example, `0#Degree/0#C4D4/` represents the motion from C4 to D4.

- Chord objects [20 for each]:

  `0#Chord/0#TCCCCCCC/`, `0#Chord/0#TEmEmEmEmEmEmEm/`, `0#Chord/0#TAmAmAmAmAmAmAm/`, `0#Chord/0#DEmEmEmEmEmEmEm/`, `0#Chord/0#DGGGGGGG/`, `0#Chord/0#DBmBmBmBmBmBmBm/`, `0#Chord/0#SDmDmDmDmDmDmDm/`, `0#Chord/0#SFFFFFFF/`, `0#Chord/0#SAmAmAmAmAmAmAm/`, `0#Chord/0#D2DmDmDmDmDmDmDm/`, `0#Chord/0#D2FFFFFFF/`.

  An object `0#Chord/0#TCCCCCCC/` represents the tonic chord `C`. There are chords that have multiple functions such as `Am`, which is tonic and subdominant; for such a chord, we give one type of object for each function.

- Objects to replace avoid notes [10]: `0#Avoid/0#Dummy/`. This object replaces an avoid note with a `Dummy` element.

- Objects to give random notes in the place of `Dummy` [10 for each]:

  `0#Avoid/0#C4/`, `0#Avoid/0#D4/`, `0#Avoid/0#E4/`, `0#Avoid/0#F4/`, `0#Avoid/0#G4/`, `0#Avoid/0#A4/`, `0#Avoid/0#B4/`, `0#Avoid/0#C5/`.

An object `0#Avoid/0#C4/` is used to replace `Dummy` with `C4`, for example.

- Objects to terminate a phrase [100]: `0#StartStop/`.
- Objects of beginning notes [2 for each]: `0#C4/`, `0#D4/`, `0#E4/`, `0#F4/`, `0#G4/`, `0#A4/`, `0#B4/`, `0#C5/`. These objects serve as "seeds." A phrase starts to grow from a seed using objects for conjunct motion.

The numbers of objects are given somewhat arbitrarily. We chose them so that a few phrases can be generated in a reasonable period of time when the description is run on our simulator.

4.3 Recombination Rules

We give a group of rules to each step of the algorithm shown at the beginning of Section 3. Each group is explained below with an example.

Making a sequence of notes. The following rule adds a next note to a sequence that ends with D4 (the first term of lhs) using a conjunct motion object (the second term of lhs); if the object is `0#Degree/0#D4E4/`, the note E4 (matched by the wildcard `<2>`) is added to the sequence (the first term of rhs).

$$\texttt{0\#<*1>D4/} + \texttt{0\#Degree/0\#D4<2>/} \to \texttt{0\#<*1>D4<2>/} + \texttt{0\#Degree/0\#D4/} \quad (1)$$

Similar rules are given for all the pitches C4 through C5. (Since our artificial chemistry permits only one occurrence of a wildcard on each side of a recombination rule, a generic rule such as "$\texttt{0\#<*1><3>/} + \texttt{0\#Degree/0\#<3><2>/} \to \cdots$" is not allowed. However, introducing syntax sugar will enable such a rule without affecting the expressive power of the artificial chemistry.)

Choosing a chord. As explained in Section 3.2, a chord for a bar is chosen according to its first note. If the note is E4, possible chords are Em, C and Am. The following three rules express it. (The notation `<1..7>` is an abbreviation for `<1><2><3><4><5><6><7>`.)

$$\texttt{0\#E4<1..7><8><9*>/} + \texttt{0\#Chord/0\#TEmEmEmEmEmEmEm/}$$
$$\to \texttt{0\#E4<1..7>/0\#TEmEmEmEmEmEmEm/} + \texttt{0\#<8><9*>/} + \texttt{0\#Chord/} \quad (2)$$

$$\texttt{0\#E4<1..7><8><9*>/} + \texttt{0\#Chord/0\#TCCCCCCC/}$$
$$\to \texttt{0\#E4<1..7>/0\#TCCCCCCC/} + \texttt{0\#<8><9*>/} + \texttt{0\#Chord/} \quad (3)$$

$$\texttt{0\#E4<1..7><8><9*>/} + \texttt{0\#Chord/0\#TAmAmAmAmAmAmAm/}$$
$$\to \texttt{0\#E4<1..7>/0\#TAmAmAmAmAmAmAm/} + \texttt{0\#<8><9*>/} + \texttt{0\#Chord/} \quad (4)$$

Each rule cuts the first eight notes of the sequence into a bar, along with adding the chord as the second line of the object (the first term of rhs).

Because Am is not only tonic but also subdominant, one more rule is given:

$$\texttt{0\#E4<1..7><8><9*>/} + \texttt{0\#Chord/0\#SAmAmAmAmAmAmAm/}$$
$$\to \texttt{0\#E4<1..7>/0\#SAmAmAmAmAmAmAm/} + \texttt{0\#<8><9*>/} + \texttt{0\#Chord/} \quad (5)$$

In similar ways, recombination rules to add chords are given for the other chords and functions.

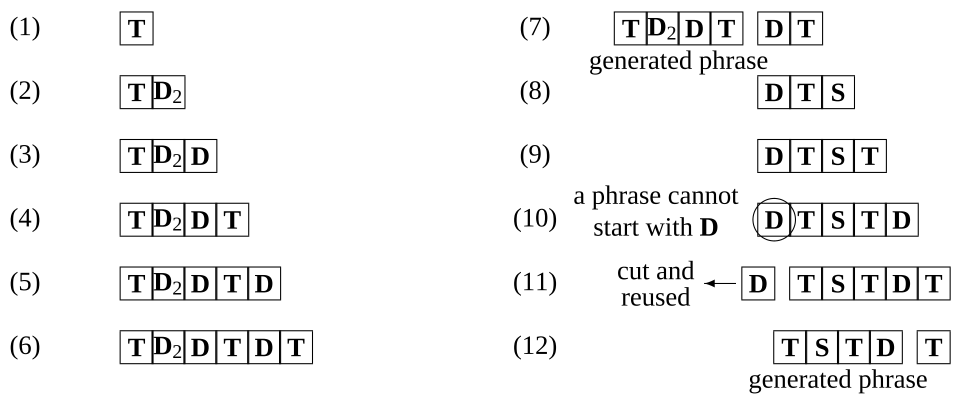


Figure 2: An example process of generating phrases.

Replacing avoid notes. Replacing an avoid note comprises two steps. The first step replaces the avoid note with a `Dummy` element. For example, the chord `C` has F4 as its avoid pitch so we give the following rule to substitute `Dummy` for F4:

$$\texttt{0\#<*1>F4<2*>/0\#<*3>C<4*>/} + \texttt{0\#Avoid/0\#Dummy/} \\ \rightarrow \texttt{0\#<*1>Dummy<2*>/0\#<*3>C<4*>/} + \texttt{0\#Avoid/0\#F4/} \quad (6)$$

Similar rules are given for the other chords. The second step replaces `Dummy` with an arbitrary note, which is performed by the rule below using an object that gives a random note (e.g., `0#Avoid/0#C4/`, matched by the second term of lhs):

$$\texttt{0\#<*1>Dummy<2*>/0\#<*3><4><5*>/} + \texttt{0\#Avoid/0\#<6>/} \\ \rightarrow \texttt{0\#<*1><6><2*>/0\#<*3><4><5*>/} + \texttt{0\#Avoid/0\#Dummy/} \quad (7)$$

Concatenating bars and obtaining phrases. As a sequence of bars is represented by an object in our design, concatenating bars and obtaining phrases are described in terms of reactions involving this type of object. An example process is illustrated in Figure 2.

A bar serves as a seed for this process (Figure 2(1)), and bars are added one by one to this seed. The rule below adds a bar with a second dominant chord to a bar sequence that ends with a tonic chord. Thus this rule implements the progression of **T**-$\mathbf{D}_2$ (the step (1) to (2) in Figure 2).

$$\texttt{0\#<*1><2..9>/0\#<*10>T<11..17>/} + \texttt{0\#<18><19><20*>/0\#D2<21*>/} \\ \rightarrow \texttt{0\#<*1><2..9><18><19><20*>/0\#<*10>T<11..17>D2<21*>/} \quad (8)$$

We give similar rules for each of the other cadence patterns (discussed in Section 3.4) **T-D**, **T-S**, **D-T**, $\mathbf{D}_2$**-D** and $\mathbf{D}_2$**-T**, which are used in the steps (1) through (6) in Figure 2.

In order to obtain a four-bar phrase that starts with a tonic chord, we give the following rule (applied in the step (7) in Figure 2).

$$\texttt{0\#<1..32><33><34*>/0\#T<35..65><66><67*>/} + \texttt{0\#StartStop/} \\ \rightarrow \texttt{0\#Start<1..32>Stop/1\#T<35..65>/} + \texttt{0\#<33><34*>/0\#<66><67*>/} \quad (9)$$

The first term of rhs represents the obtained four-bar phrase terminated with `Start` and `Stop`, resulting such an object as shown in Figure 1(a).

The remaining sequence of bars (represented by the second term of rhs) continues growing (the steps (8) through (10)). However, this sequence may start with a non-tonic chord. In such a case, even if

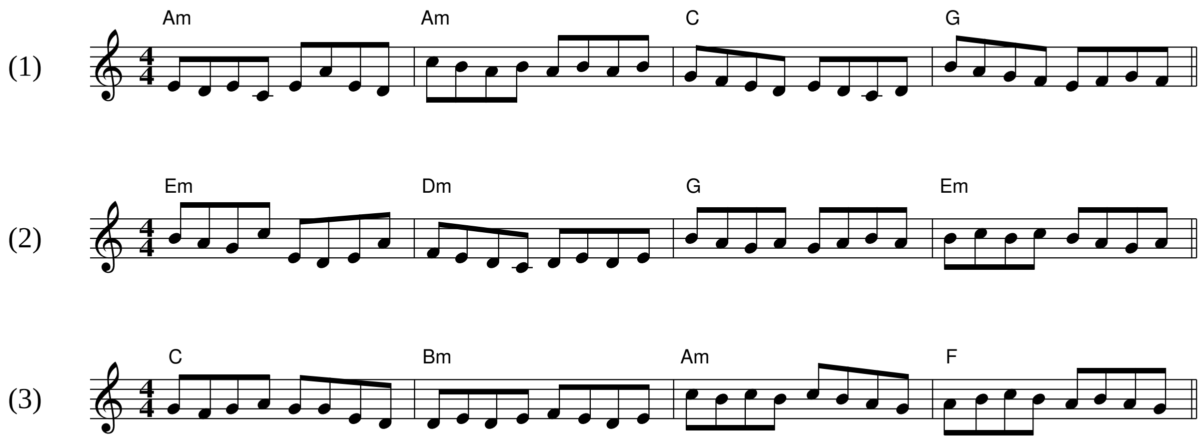


Figure 3: Some phrases generated by the artificial chemistry.

the sequence gets longer than four bars (e.g., Figure 2(10)), no phrase starting with a tonic chord can be obtained from the head of this sequence. In order to recover from this situation, we add rules to cut off the first bar from a phrase if the bar has a non-tonic chord. An example rule is shown below, which detaches the first bar if it has a dominant chord (Figure 2(11)).

```
0#<1..8><9><10*>/0#D<11..17><18><19*>/
                          → 0#<1..8>/0#D<11..17>/ + 0#<9><10*>/0#<18><19*>/   (10)
```

The detached bar will be reused to make another phrase. We give similar rules for the other non-tonic functions $\mathbf{D}_2$ and **S**.

5 Simulation and Results

We gave the description of the system illustrated above to our simulator of the artificial chemistry and performed simulation. The current simulator is implemented on the Cocoa Framework / Mac OS X using Objective-C. It employs a reactor algorithm based on random collision. Although the algorithm is qualitatively sound (i.e., the simulator follows one possible computation path among from all the possibilities represented by the given nondeterministic algorithm), it still lacks sufficient theoretical foundation for simulating quantitative behaviour. We therefore show only the products of the runs as preliminary results, abstaining from time-series analysis of their behaviour. Such analysis, along with establishing a chemically plausible algorithm and improving the simulator, is our future work.

Once an object representing a four-bar phrase is terminated by the elements `Start` and `Stop` by the application of Rule 9, the object is no longer recombined by any rule. We collect objects of this form as final products since the system has no terminating condition. A typical run generates three to five phrases in approximately one hour on Power PC G5 1.8GHz. The phrase shown in Figure 1 is a generated phrase, and some other phrases are shown in Figure 3. While melodies are generated based on conjunct motion, each phrase shown in the Figures has a jump in the melody. This is because avoid notes are replaced with random notes.

6 Discussion

The process of music composition given in Section 3 represents a vast number of musical phrases that observe the following style of music: (1) the melody of a bar is smooth in the sense that the movement is based on conjunct motion; (2) a bar has little dissonance caused by avoid notes; and (3) the chord progression is natural in the sense it follows the cadence rules. Since there are multiple choices in most of the compositional steps, the process is expressed as a nondeterministic algorithm. The algorithm is then implemented in the artificial chemistry in an straightforward manner: the recombination rules correspond to the steps of the algorithm, and intermediate data (i.e., computational states of the nondeterministic algorithm) are represented as objects. The inherent nondeterminism of the artificial chemistry enabled the algorithm to be implemented straightforwardly.

This nondeterministic approach to composing music has the following benefits over a deterministic approach.

- In almost every step of composing musical phrases, there are multiple choices such as choosing a note or a chord. If this is achieved by a deterministic algorithm, it must utilise randomness explicitly to make a choice. In our approach this property is already built in the nondeterminism of the framework, so the designer of rules and objects needs not care about it.
- To make a multiple-bar phrase, a process has to generate a bar with a chord with an appropriate function. For example, if the process has a sequence of bars that ends with a dominant bar, the next bar must be a tonic bar. A deterministic algorithm must deliberately generate a tonic bar. In contrast, our algorithm can choose an appropriate bar from a set of bars already generated (or going to be generated).
- Our approach can produce phrases in parallel using the inherent nondeterminism of the algorithm. A deterministic algorithm must be explicitly parallelised if it is to run on a parallel platform.

Generating musical phrases can be viewed as a search problem, and genetic algorithms (GAs) have been applied to this problem. GA methods can be categorised into four approaches based on types of fitness functions, namely, deterministic, formalistic, user-determined and neural [3]. Among them, similar to ours is the formalistic approach, whose fitness functions give high fitness values to musical phrases that adhere to existing rules of musical theory. The main difference is as follows. In each generation in a GA run, some individuals are fit and others unfit; in other words, some of them satisfy the given formalistic requirements and others do not. In our approach, the nondeterministic algorithm is designed so that every generated phrase satisfies the formalistic requirements; we need no “fitness function” to evaluate how good (i.e., how well abiding by the formalism) a generated phrase is. Moreover, in GA methods, unfit individuals are simply discarded if the method uses the standard selection technique, while our algorithm has steps to improve or reuse them: avoid notes are replaced to make better harmony (note that it is hard to replace them by random mutation), and a dominant bar at the top of a phrase is cut and reused. From this viewpoint a nondeterministic algorithm is more flexible than GA. The main limitation of our approach is that the designer must know beforehand an algorithm for composition, or *how* to compose music. On the other hand, a fitness function of GA can represent *what* music is desired. We do not consider this as a shortcoming of our approach; they can rather be combined. A possible scenario is as follows: first, an artificial chemistry generates a number of phrases that adhere to a specific musical formalism; then they are fed to GA as the initial population to obtain better phrases.

Beyls [1] proposed a model for generating melodies based on simulated molecular collisions in combination with GA. The distances between molecules are interpreted as musical events. The user can change the rules for collisions interactively. The purpose of the system is to observe the emergence of unexpected musical structure; the system would be hard to be applied to create musical phrases that conform to an arbitrary musical style. In contrast, our approach is more amenable to styles since a nondeterministic algorithm allows a great degree of control.

Cellular automata are also used to implement systems that generate music. One of such systems is CAMUS [5], which uses two kinds of cellular automata to produce phrases of triads. The coordinates of an active cell decide the intervals of a triad; the root pitch of the triad is decided from parameters given by the user. Active cells change as the cellular automata run, by which a sequence of triads is generated. The aim of CAMUS is to create new kinds of music; ours is to generate music in a certain style, which is difficult to implement simply with limited parameter control such as that available in CAMUS.

SWARMUSIC [2] is a swarm system that generates music. The emphasis of this system is on real-time improvisation and interaction, not algorithmic composition of music, which we aim in this study.

7 Conclusion

In this paper, we implemented a simple music composing algorithm as a system in our artificial chemistry, and showed that it can produce musical phrases without human intervention. The nondeterministic algorithm was straightforwardly described as rules in the artificial chemistry; the simulator acted as a processor of the algorithm. It suggests that an artificial chemistry may be a useful platform for describing and processing nondeterministic algorithms due to its inherent nondeterminism. Furthermore, thinking of the compatibility with molecular computing, artificial chemistries would possibly serve as an intermediary between molecular computing and nondeterministic algorithms.

References

[1] Peter Beyls. A molecular collision model of musical interaction. In Celestino Soddu, editor, *Proceedings of the 8th International Conference on Generative Art (GA2005)*, pages 375–386, Milan, 2005. AleaDesign.

[2] Tim Blackwell and Peter Bentley. Improvised music with swarms. In *Proceedings of the 2002 Congress on Evolutionary Computation*, volume 2, pages 1462–1467, Piscataway, NJ, 2002. IEEE Press.

[3] Anthony Burton and Tanya Vladimirova. Generation of musical sequences with genetic techniques. *Computer Music Journal*, 23(4):59–73, 1999.

[4] Peter Dittrich, Jens Ziegler, and Wolfgang Banzhaf. Artificial chemistries — a review. *Artificial Life*, 7(3):225–275, 2001.

[5] Eduardo Miranda. On the music of emergent behavior: What can evolutionary computation bring to the musician? *Leonardo*, 36(1):55–59, 2003.

[6] Yoshikazu Suzuki and Kazuto Tominaga. Describing metabolic pathways using an artificial chemistry based on pattern matching and recombination. In Masanori Sugisaka and Hiroshi Tanaka, editors, *Proceedings of the 11th International Symposium on Artificial Life and Robotics (AROB '06)*, 2006. Published as CD-ROM.

[7] Kazuto Tominaga. A formal model based on affinity among elements for describing behavior of complex systems. Technical Report UIUCDCS-R-2004-2413, Department of Computer Science, University of Illinois at Urbana-Champaign, March 2004.

[8] Kazuto Tominaga. Describing protein synthesis and a cell cycle of an imaginary cell using a simple artificial chemistry. In *Proceedings of the Workshop on Artificial Chemistry and Its Applications, part of the 8th European Conference on Artificial Life (ECAL) 2005*, 2005. Published as CD-ROM.

[9] Kazuto Tominaga, Keiji Kobayashi, Tooru Watanabe, Kazumasa Koizumi, and Koji Kishi. An approach to constructing qualitative models in computational cell biology using an artificial chemistry based on pattern matching and recombination. In *Proceedings of the 10th International Symposium on the Simulation and Synthesis of Living Systems (ALIFE X)*, 2006. In press.

[10] Tooru Watanabe, Keiji Kobayashi, Masaki Nakamura, Koji Kishi, Mitsuyoshi Kazuno, and Kazuto Tominaga. Describing DNA automata using an artificial chemistry based on pattern matching and recombination. In Hussein Abbass, Terry Bossomaier, and Janet Wiles, editors, *Recent Advances in Artificial Life*, volume 3 of *Advances in Natural Computation*. World Scientific, Singapore, 2005.

Category Theoretical Formalization of Autopoieis from Perspective of Distinction between Organization and Structure

Tatsuya Nomura

Department of Media Informatics, Ryukoku University
1–5, Yokotani, Seta–ohe–cho, Otsu, Shiga 520–2194, Japan
nomura@rins.ryukoku.ac.jp

Abstract

Since the concept of autopoiesis was proposed as a model of minimal living systems by Maturana and Varela, and applied to social systems by Luhmann, there has been only a few mathematically strict models to represent the characteristics of it because of its difficulty for interpretation. In order to explore the validity of this concept, this paper proposes a more general formal description of autopoiesis based on the theory of category. This paper focuses on the distinction of organizations and structures, and then discusses its implications and problems on formalization of autopoiesis.

1 Introduction

Autopoiesis gives a framework in which a system exists as an organism through physical and chemical processes, based on the assumption that organisms are machinery [7]. According to the original definition of it by Maturana and Varela, an autopoietic system is one that continuously produces the components that specify it, while at the same time realizing itself to be a concrete unity in space and time; this makes the network of production of components possible. An autopoietic system is organized as a network of processes of production of components, where these components:

1. continuously regenerate and realize the network that produces them, and
2. constitute the system as a distinguishable unity in the domain in which they exist.

The characteristics of autopoietic systems Maturana gives are as follows:

Autonomy: Autopoietic machinery integrates various changes into the maintenance of its organization. A car, the representative example of non–autopoietic systems, does not have any autonomy.

Individuality: Autopoietic machinery has its identity independent of mutual actions between it and external observers, by repeatedly reproducing and maintaining the organization. The identity of a non–autopoietic system is dependent on external observers and such a system does not have any individuality.

Self–Determination of the Boundary of the System: Autopoietic machinery determines its boundary through the self–reproduction processes. Since the boundaries of non–autopoietic systems are determined by external observers, self–determination of the boundaries does not apply to them.

Absence of Input and Output in the System: Even if a stimulus independent of an autopoietic machine causes continuous changes in the machine, these changes are subordinate to the maintenance of the organization which specifies the machine. Thus, the relation between the stimulus and the changes lies in the area of observation, and not in the organization.

This system theory has been applied to a variety of fields including sociology [5]. However, there has been only a few mathematically strict models to represent the characteristics of it because of its difficulty for interpretation. McMullin has studied a computational model of autopoiesis as 2–D biological cells [9]. Bourgine and Stewart proposed a mathematical formalization of autopoiesis as random dynamical systems and explored the relationships between autopoiesis and cognitive systems [2]. Letelier, et. al., suggested the relationships between autopoiesis and metabolism–repair systems [6], which is an abstract mathematical model of biological cells proposed by Rosen [13]. Nomura also proposed a mathematical model of autopoiesis based on Rosen's system [10]. These models vary from abstract algebraic formalization to computational models based on artificial chemistry.

In order to explore the validity of autopoiesis more deeply, this paper reconsiders necessary conditions for modeling characteristics of autopoiesis based on Kawamoto's theory [4], and discusses a problem of the above existing models of autopoiesis. On these consideration and discussion, this paper uses the category theoretic formalization of autopoiesis, which was proposed by Nomura [11, 12] to clarify whether autopoiesis can really be represented within the conventional mathematical frameworks.

2 Distinction between Organization and Structure

In Japan, Kawamoto has continued his own development of autopoiesis [4]. Kawamoto's theory focuses on circular relations of components and the network of production processes of components.

Kawamoto's important claims are as follows: an autopoietic system is a network consisting of relations between production processes of components. This network produces components of the system, and the components exist in physical spaces. Then, the system exists only if the components reproduce the network of production processes. The structure of the system is a realization of the system through the operation of the system in the physical space, and the organization of the system is a form of the network. The organization is functionally specified, although the structure is realized in the physical space.

The above claims by Kawamoto have an important implication. The organization of a system differs from the structure since they exist in different levels. This distinction is mentioned in Maturana and Varela's original literature [8]. Figure 1 shows this aspect. The distinction between organizations and structures in an autopoietic system can be interpreted as a distinction between categories on which the organization and structure of a autopoietic system are defined in a mathematical formalization of it.

This distinction is suggested from another formal perspective.

Rosen compared machine systems with living systems to clarify the difference between them,

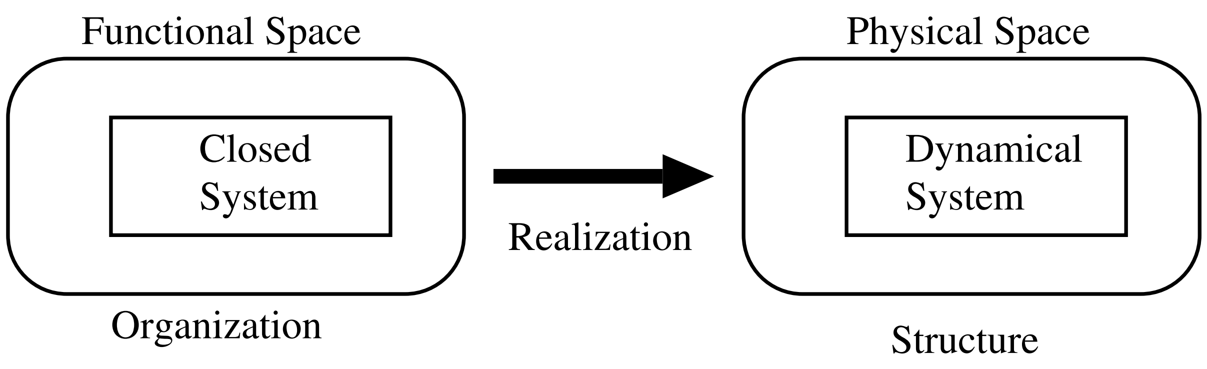


Figure 1: Aspect of Autopoiesis based on Distinction between Organization and Structure

based on the relationship among components through entailment [13]. In other words, he focused his attention on where the function of each component results from in the sense of Aristotle's four causal categories, that is, material cause, efficient cause, formal cause, and final cause. As a result, Rosen claimed that a material system is an organism if and only if it is closed to efficient causation. Furthermore, Rosen suggested that systems closed under efficient cause cannot be described with their states because they lead to infinite regress [13].

Nomura proposed a category theoretic formalization of autopoiesis under the assumption that closure under entailment or production is a necessary condition for a system to be autopoietic because the components reproduce themselves in the system [11, 12]. Although this formalization showed the possibility of constructing systems closed under entailment in specific categories, these categories had to satisfy the condition that operands coincide with operators ($X \simeq X^X$). Although Soto–Andrade and Varela provided a category satisfying this condition (the category of partially ordered sets and continuous monotone maps with special conditions), this category is very special [14].

The above two works have an important implication. If circular relations between components and their production process network are closed under entailment, this closedness may be hard to be formalized in general state spaces. On the other hand, the structure of an autopoietic system must be realized in a state space as a physical one. These implications suggest the distinction between organizations and structures in formalization of autopoiesis.

However, there is no general model of autopoiesis reflecting this distinction. The existing computational models and dynamical system models are defined on state spaces specific for them. In other words, they specify structures of systems as relations between elements of the systems, and have no explicit formalization of the organizations.

On the other hand, Nomura's category theoretic model [11, 12] represents only the aspect of closedness in organizations, and lacks the structures in autopoiesis. The next section proposes a category theoretic formalization of autopoiesis involving the distinction between organizations and structures, by complementing this lack.

3 Category Theoretic Model of Distinction between Organizations and Structures

3.1 Theory of Category

Category theory is an algebraic framework to abstractly handle the collection of mathematical objects having some specific properties, such as “the collection of all groups”, “the collection of all sets”, “the collection of all topological spaces”, “the collection of differential manifolds”, and so on [1]. In this framework, an individual space or set is dealt with as an object, and a function or map from an object to another object is dealt with as a morphism corresponding to an arc between them. Thus, the inner structures of any object and morphism are reduced, and pure relations of morphisms between are focused on. This can make it possible to investigate what category of mathematical objects a specific relation between objects (for example, closed relations between objects and morphisms) is satisfied in.

In addition, category theory can deal with relations of categories themselves as functors. This can make it possible investigate relations between a specific category and general ones such as state spaces.

As mentioned in the previous section, the organization of an autopoietic system should be formalized as a network of components and production processes, closed under entailment. Then, the structure of the system should be realized in a state space. The proposal in this paper is that the organization is formalized in a specific category, the structure is formalized in the category of general state spaces, and realization from the organization to the structure in the autopoietic system is modeled by a functor between the categories.

3.2 Completely Closed Systems as Organizations

This paper shows a formalization using “completely closed systems” as an example of organization [11, 12]. Although there are other closed systems to be considered as organizations, this paper focuses on this simple type of systems to provide with easier interpretation.

We assume that an abstract category $\mathcal{C}$ has a final object 1 and product object $A \times B$ for any pair of object A and B. The category of all sets is an example of this category. Moreover, we describe the set of morphisms from A to B as $H_{\mathcal{C}}(A, B)$ for any pair of objects A and B. A element of $H_{\mathcal{C}}(1, X)$ is called a morphic point on X. For a morphism $f \in H_{\mathcal{C}}(X, X)$ and a morphic point x on X, x is called a fixed point of f iff $f \circ x = x$ ($\circ$ means composition of morphisms) [14]. Morphic points and fixed points are respectively abstraction of elements of a set and fixed points of maps in the category of sets.

When there exists the power object Y^X for objects X and Y (that is, the functor $\cdot \times X$ on $\mathcal{C}$ has the right adjoint functor $\cdot^X$ for X), note that there is a natural one–to–one correspondence between $H_{\mathcal{C}}(Z \times X, Y)$ and $H_{\mathcal{C}}(Z, Y^X)$ for any objects X, Y, Z satisfying the diagram in the left half of figure 2. Thus, there is a natural one–to–one correspondence between morphic points on Y^X and morphisms from X to Y satisfying the diagram in the right half of figure 2 [15].

When components in a system are not only operands but also operators, the easiest method for representing this aspect is the assumption of existence of an isomorphism from the space of operands to the space of operators [3].

Now, we assume an object X with powers and an isomorphism $f : X \simeq X^X$ in $\mathcal{C}$. Then, there uniquely exists a morphic point p on $(X^X)^X$ corresponding to f in the above sense, that

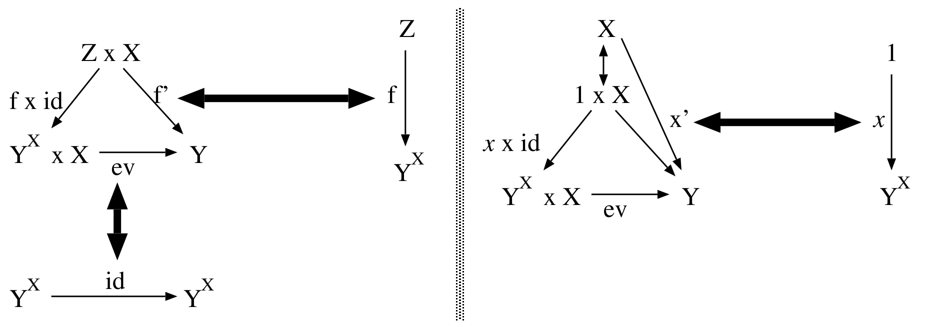


Figure 2: Natural One–To–One Correspondence between $H_{\mathcal{C}}(Z \times X, Y)$ and $H_{\mathcal{C}}(Z, Y^X)$

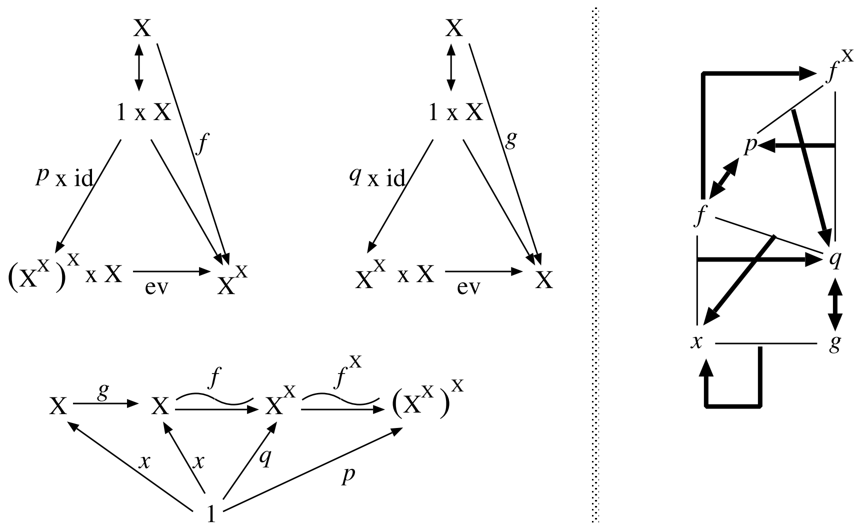


Figure 3: The Diagrams of a Completely Closed System and the Entailment Relations

is , $p' = f$. Since the morphism from X^X to $(X^X)^X$ entailed by the functor $\cdot^X$, f^X, is also isomorphic, there uniquely exists a morphic point q on X^X such that $f^X \circ q = p$. We can consider that p and q entail each other by f^X. Furthermore, there uniquely exists a morphic point x on X such that $f \circ x = q$ because f is isomorphic. Since we can consider that x and q entail each other by f, and f and p entail each other by the natural correspondence, the system consisting of x, q, p, f, and f^X is completely closed under entailment. Moreover, if x is a fixed point of $g : X \to X$ naturally corresponding to q, that is, $g \circ x = x$, we can consider that x entails itself by g. Figure 3 shows the diagrams of this completely closed system and the entailment relations.

3.3 Structures Induced by Completely Closed Systems

Here, we consider the formalization of structures induced by the organization mentioned in the previous section as follows.

We assume a family of categories $\{C_\lambda\}_\lambda$, and that each C_λ includes a completely closed system $\{X_\lambda, x_\lambda \in H_{C_\lambda}(1, X_\lambda), f_\lambda \in H_{C_\lambda}(X_\lambda, {X_\lambda}^{X_\lambda}), q_\lambda \in H_{C_\lambda}(1, {X_\lambda}^{X_\lambda}), p_\lambda \in H_{C_\lambda}(1, \left({X_\lambda}^{X_\lambda}\right)^{X_\lambda}), g_\lambda \in H_{C_\lambda}(X_\lambda, X_\lambda)\}$. Moreover, we assume another category D. Here, it is assumed that D is the category consisting of state spaces and maps between them, or its subcategory.

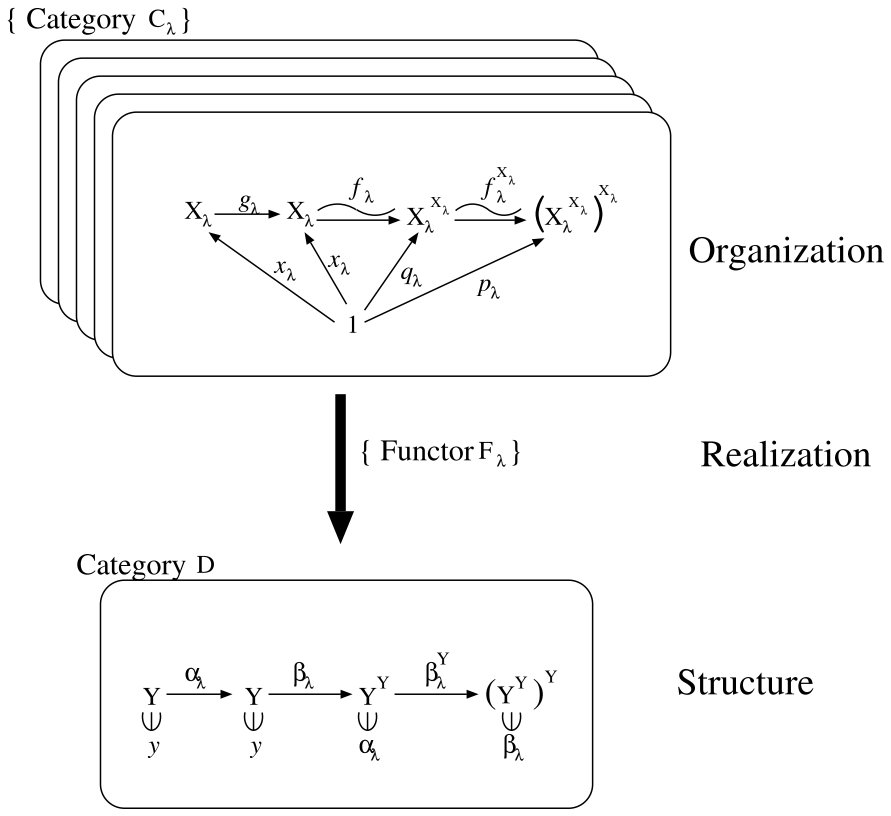

{ Category C_λ }
$X_\lambda \xrightarrow{g_\lambda} X_\lambda \xrightarrow{f_\lambda} X_\lambda^{X_\lambda} \xrightarrow{f_\lambda^{X_\lambda}} (X_\lambda^{X_\lambda})^{X_\lambda}$
x_λ, x_λ, q_λ, p_λ, 1
Organization
{ Functor F_λ }
Realization
Category D
$Y \xrightarrow{\alpha_\lambda} Y \xrightarrow{\beta_\lambda} Y^Y \xrightarrow{\beta_\lambda^Y} (Y^Y)^Y$
y, y, α_λ, β_λ
Structure

Figure 4: Organization of an Autopoietic System and Its Realized Structure

Now, we assume a family of functors $\{F_\lambda : C_\lambda \to D\}_\lambda$ such that $F_\lambda(1_{C_\lambda}) = 1_D$, $Y = F_\lambda(X_\lambda)$, $y = F_\lambda(x_\lambda) \in H_D(1, Y) = Y$ for all λ, and $\alpha_\lambda = F_\lambda(g_\lambda) \in H_D(Y,Y) = H(1, Y^Y) = Y^Y$ for any λ (here, we can regard $H_D(1,Y) = Y$ and $H_D(Z,Y) = H_D(1, Y^Z) = Y^Z$ since D is the category of state spaces). Note that $\alpha_\lambda(y) = \alpha_\lambda \circ y = F_\lambda(g_\lambda) \circ F_\lambda(x_\lambda) = F_\lambda(g_\lambda \circ x_\lambda) = F_\lambda(x_\lambda) = y$ for any λ, that is, y is a fixed point of α_λ.

We propose that the family of the completely closed systems $\{\{X_\lambda, x_\lambda, f_\lambda, q_\lambda, p_\lambda, g_\lambda\}\}_\lambda$ is an organization of an autopoietic system and $\{Y, y, \{\alpha_\lambda\}_\lambda\}$ is its structure realized on the category D through the family of the functors $\{F_\lambda\}$ if for any λ one of the following conditions is satisfied:

1. $^\exists\beta_\lambda \in H_D(Y, Y^Y) = H_D\left(1, (Y^Y)^Y\right) = (Y^Y)^Y$ *s.t.*, $\beta_\lambda(y) = \alpha_\lambda$, ${\beta_\lambda}^Y(\alpha_\lambda) = \beta_\lambda$
2. $^\exists\beta_\lambda \in H_D(Y, Y^Y)$, λ_1, λ_2, and $\beta_{\lambda_2} \in H_D(Y, Y^Y)$ *s.t.*, $\beta_\lambda(y) = \alpha_{\lambda_1}$, $\beta_\lambda = {\beta_{\lambda_2}}^Y(\alpha_{\lambda_2})$

The above relationship between the organization and structure represents the aspect that the structure is entailed repeatedly within the organization. Figure 4 shows these organization and realized structure.

4 Discussion

This section discusses some implications of the formalization of autopoiesis proposed in the previous section, and some problems of it.

4.1 Implications

The proposed formalization of autopoiesis explicitly represents a distinction between organizations and structures. The following four facts can be implied from this representation.

First, the organization is static and closed under entailment between morphic points and morphisms. This implies the aspect of autopoiesis that the network consisting of relations between production processes of components is reproduced by the components.

Second, by distinguishing the category on which the structure is realized from the categories on which the organization is defined, a kind of dynamics in the structure is implied. On this dynamics, a part of the structure α_λ is dealt with. The first condition of structure in the previous section means that α_λ is dynamically maintained and the structure is closed by the existence of β_λ. The second condition of structure in the previous section means that α_λ is dynamically changed within the organization, that is, change of the structure under the unique organization.

Third, by introducing realization as a family of functors from the categories of organization to the category of state spaces, non–uniqueness of structures for the organization is represented. In other words, for the same organization $\{\{X_\lambda, x_\lambda, f_\lambda, q_\lambda, p_\lambda, g_\lambda\}\}_\lambda$, the existence of another structure $\{Y', y', \{\alpha'_\lambda\}_\lambda\}$ and its realization $\{F'_\lambda\}$ are implied. This suggests that a structure of autopoiesis having an organization based some physical materials can be realized based on other materials.

Fourth, the formalization in the paper uses completely closed systems as a simpler example of organization. Of course, closed systems of organizations are not limited to completely closed systems [12]. Thus, the proposed formalization implies a variety of organizations, structures, and realization.

4.2 Problems

On the other hand, the proposed formalization of autopoiesis has the following problems.

The proposed formalization assumes that organizations are formalized on categories permitting the existence of an isomorphism from the space of operands to the space of operators, in prior to state spaces on which the structures are realized. Then, change of the structures is fixed within the organizations. This is anticipation of the issues in a sense.

This is critical when we consider whether a system can be identified as autopoiesis by observers who can only see the structure on a state space. In order for the observers to be able to identify the system as autopoiesis, they must be able to find the organizations that cannot be formalized on the state space, and the categories of functional spaces on which the organizations are closed in the sense that the network consisting of relations between production processes of components is reproduced by the components. When these observers assume the organizations based on only the structures, however, there is arbitrariness in this assumption since the proposed model does not include any specification of organizations from structures. In this sense, the proposed formalization of autopoiesis may not be autopoiesis itself but just a cognitive model of the way in which these observers identify the system as autopoiesis.

This critical problem is caused by explicit distinction of organizations and structures, and closedness of organizations. As far as closedness of organizations is assumed, organizations are hard to be found in state spaces on which structures are realized. Thus, another category in an abstract level is necessary. As one of ways to refine the proposed formalization, we consider its application to the existing computational and dynamical systems models of autopoiesis, that is, investigation of categories of organization assuming these models as structures. By this application, we can find whether the proposed formalization can discriminate between autopoietic and non–autopoietic systems.

5 Summary

This paper focused on the distinction of organizations and structures in autopoiesis reconsidering necessary conditions for modeling characteristics of autopoiesis based on Kawamoto's theory [4]. Then, a general formalization of autopoesis based on category theory was proposed while explicitly representing the distinction of organizations and structures. In addition, some implications and problems were discussed.

As an important future work, we consider application of the proposed framework to real systems including biological, mental, and social systems. This can allow us to investigate whether the proposed framework is useful to clarify the difference between autopoietic and non–autopoietic systems at an abstract mathematical level.

References

[1] Borceux, F. (1994). *Handbook of Categorical Algebra 1: Basic Category Theory.* Cambridge University Press.

[2] Bourgine, P. and Stewart, J. (2004). Autopoiesis and cognition. *Artificial Life*, 10(3):327–346.

[3] Kampis, G. (1991). *Self–Modifying Systems in Biology and Cognitive Science: A New Framework for Synamics, Information, and Complexity.* Pergamon Press.

[4] Kawamoto, H. (1995). *Autopoiesis: The Third Generation System.* Seido–sha Publishers. (in Japanese).

[5] Kneer, G. and Nassehi, A. (1993). *Niklas Luhmanns Theorie Sozialer Systeme.* Wilhelm Fink Verlag. (Japanese Edition: Tateno, T., et.al., (1995). Shinsensha.).

[6] Letelier, J. C., Soto-Andrade, J., Abarzúa, F. G., Cornish-Bowden, A., and Cárdenas, M. L. (2004). Metabolic closure in (M, R) systems. In *Proc. 9th Int. Conf. Simulation and Synthesis of Living Systems (ALIFE9)*, pages 450–461.

[7] Maturana, H. R. and Varela, F. J. (1980). *Autopoiesis and Cognition: The Realization of the Living.* D. Reidel Publishing. (Japanese Edition: Kawamoto, H. (1991). Kokubun–sha Publishers.).

[8] Maturana, H. R. and Varela, F. J. (1987). *The Tree of Knowledge.* Shambala Publications. (Japanese edition: Suga, K. (1987). Asahi Publications).

[9] McMullin, B. (2004). Thirty years of computational autopoiesis. *Artificial Life*, 10(3):277–296.

[10] Nomura, T. (1997). An Attempt for Description of Quasi–Autopoietic Systems Using Metabolism–Repair Systems. In *Proc. 4th European Conf. Artificial Life (ECAL'97)*, pages 48–56. MIT Press.

[11] Nomura, T. (2001). Formal description of autopoiesis based on the theory of category. In *Proc. 6th European Conf. Artificial Life (ECAL'01)*, pages 700–703.

[12] Nomura, T. (2002). Formal description of autopoiesis for analytic models of life and social systems. In *Proc. 8th Int. Conf. Artificial Life (ALIFE VIII),*, pages 15–18.

[13] Rosen, R. (1991). *LIFE ITSELF.* Columbia University Press.

[14] Soto-Andrade, J. and Varela, F. J. (1984). Self–reference and fixed points: A discussion and an extension of Lawvere's theorem. *Acta Applicandae Mathematicae*, 2:1–19.

[15] Takeuchi, G. (1978). *Sheaf, Category, and Topos.* Nihon Hyoron–sha. (in Japanese).

Computing Substrates and Life

Soichiro Tsuda[1], Klaus-Peter Zauner[2], and Yukio-Pegio Gunji[1]

[1] Graduate School of Science and Technology
Kobe University, Nada, Kobe 657-8501, Japan
026d874n@stu.kobe-u.ac.jp, yukio@kobe-u.ac.jp, fax: +81-78-803-5759

[2] School of Electronics and Computer Science
University of Southampton, SO17 1BJ, United Kingdom
kpz@ecs.soton.ac.uk, fax: +44-23-8059-2865

Abstract

Alive matter distinguishes itself from inanimate matter by actively maintaining a high degree of inhomogenous organisation. Information processing is quintessential to this capability. The present paper inquires into the degree to which the information processing aspect of living systems can be abstracted from the physical medium of its implementation. Information processing serving to sustain the complex organisation of a living system faces both the harsh reality of real-time requirements and severe constraints on energy and material that can be expended on the task. This issue is of interest for the potential scope of Artificial Life and its interaction with Synthetic Biology. It is pertinent also for information technology. With regard to the latter aspect, the use of a living cell in a robot control architecture is considered.

1 Life as information process

The difference between the living and non-living is not, as once supposed, a material difference or a difference in the applicable laws of nature [1]. Now, life appears to be delimited only by a peculiar organisation of the very same matter that forms the remaining non-living universe. This organisation can be sustained only by active maintenance which in turn necessitates the processing of information. As a consequence, life without computation is inconceivable.

This is true down to the simplest organisms and even their molecular constituents. The macromolecules that underly the structure and function of living systems are not simply products of chemical reactions; they are individually assembled in sophisticated and tightly controlled production processes with fine grained quality control mechanisms—all of which require computation. Endowed with an essential information processing capability, organisms recruited it and extended it for other tasks, most prominently for acquiring nutrients, for avoiding hazards, and for reproduction.

But if one attempts the implementation of life-like artificial devices then the discrepancy between formal computation of practicable complexity and the real-time requirements in an open physical world becomes all too apparent. The question arises whether the intertwining of information processing and material processes innate to organisms may confer computational capabilities that in practice surpass conventional computing methods?

1.1 Abstract computation

Information and operations on it can be described independent of any physical implementation. This approach turned out to be quite fruitful. Hartley derived the familiar measure of information from symbol frequencies and thus deliberately removed the physical information carrier from the picture [2]. Similarly the formalisation of a human computer by Turing [3] enabled the study of computation independent of an actual realisation. Turing formulated his abstraction to show that problems do exist that cannot be solved by a computer, even if no constraints are placed on available storage space and the time it will take to arrive at a result. His abstraction also entailed a simple but general machine model for computation which subsequently turned out to be equivalent to several other formalisations of computation (in particular also equivalent to recursive functions) and was highly influential in the field of theoretical computer science. It is now generally believed that anything that in principle can be computed can also be computed on Turing's machine [4]. In fact, the mathematical notion of what is "effectively computable" refers now to the class of problems that can be solved by Turing's machine model. Such machines, capable of carrying out any computation, are called universal. It is worth emphasising that a universal machine is always a hypothetical construct—a machine that would be restricted to a finite memory cannot be universal.

Interestingly, for a machine to be universal it does not require a sophisticated mechanism nor a complex architecture. A processor with an instruction set of only the two commands 'increment' and 'decrement with conditional jump' coupled to an unlimited random access memory would be universal in the above sense. Or, for instance, closer to Turing's original formulation, a finite state automaton with only seven states and a capability to manipulate symbols in a sequential access memory is sufficient to be universal [4]. Accordingly, from the standpoint of what is in principle computable, many information processing systems have essentially the same power. Thus, no matter how much more complex the architecture of an information processor is compared to the aforementioned simple universal machines, it will not be able to perform a computation that could not also be performed by any of the simple universal machines.

After what has been stated regarding the equivalence of information processors it is reasonable to suppose that the computation requisite for life falls within the realm of the universal machines. It should therefore in principle be possible to abstract the information processing aspect of living matter by means of a universal computational model. This would seem to indicate that it is possible to satisfactorily capture the information processing aspect of living systems by a formal implementation, for instance, on a general purpose digital computer. But, as a matter of fact, attempts to replicate the essence of life in abstracted information processes have not resulted in convincing demonstrations of life-like phenomena. Given that Biology hardly ever takes explicit heed of the role information processing plays in the alive state of matter, a partial explanation of this may be that the processes themselves are not sufficiently understood to be formalised. Another explanation, however, arises from the possibility that the physical substrate that implements the computation is of greater relevance than the abstraction outlined above would permit. It is the latter possibility on which we will focus for the remainder of this paper.

1.2 Real computation

The previous section delineated a picture in which all reasonably complete computing machines have the same ultimate theoretical capability. This finding hinges on ignoring requirements in memory space and execution time. The picture changes radically if not hypothetical formal machines, but practical devices are concerned. Then computability is constrained by physical dynamics and realistic resource limitations.

It is possible to estimate the ultimate limitation of physically feasible computing from the speed of light as limit for signal distribution and the constraint Heisenberg's uncertainty principle places on discerning system states [5]. Of more immediate interest are the limitations that arise from the need of any real computation process to represent information by physical degrees of freedom. Accordingly, over the course of a computation, for every change in abstract information there has to be a corresponding change of the physical state of the hardware that implements the computation. The course of computation restricts at each stage the permissible physical states of the hardware to a subset of its possible states with concomitant thermodynamic constraints. As a result of the thermodynamic effects energy has to be expended to process information, in particular there is a fundamental minimum of energy that is required for state preparation [6].

Careful analysis revealed that it is possible to trade energy consumption, reliability, and computing speed against each other [7, 8]. The energy cost of logic operations can in principle be made arbitrarily small at the expense of speed. However, if the device would be required to successfully perform a computation, the speed for this computation cannot be arbitrarily small because the physical structure of the computing device itself will degrade over time and thus there will be minimum speed required to allow for completing the computation [9]. Consequently, there will also be a minimum energy that is necessary for driving the computation towards completion.

The above consideration pertains to a computer prepared for a computation or for repeating a computational cycle; the state preparation can also be viewed as resetting the machine. Hanson rightly draws attention to the fact that a robot interacting continously with a changing envrionment will face additional constraints on the tradeoff between energy, reliability, and speed [10]. The minimal speed of computation needs to be adequate to respond in real-time to the challenges posed by the environment, and moreover, interaction with an unpredictably changing envrionment reduces reversibility and incurs an additional cost for state preparation to delete outdated information.

Living systems, as previously stated, require computation not only to interact with their environment but also, more fundamentally, to actively maintain the intricate structure corresponding to the living state. To avoid thermodynamic equilibration a living system has to continously process the insults imposed on its organisation by the external and internal environment and take rather detailed control of its microstate to pilot it within the set of states compatible with life [11, pp.14–32]. The need for circuiting states incompatible with life places a lower bound on speed and reliability for the computation necessary to dissipate perturbations. Owing to this, computational efficiency is important to a living system and, arguably, could even be a limiting factor for its complexity. If we adopt the above perspective and the importance it assigns to information processing for the living state, it is worthwhile to inquire into the principles of natural information processing architectures.

2 The role of the physical substrate

A computer is a system that starts from a state encoding specific information and follows the laws of nature to arrive in a state that can be interpreted as information derived from the starting state (cf. [12]). This general definition encompasses any physical system as an extreme case, because any system could be viewed as computing its own behaviour. Of course, this is a trivial form of information processing as it eliminates any freedom with regard to the representation of information and its processing. This form of processing is very limited and highly specialised. But it is also highly efficient with regard to the amount of matter and time required. The other extreme within the above definition is occupied by the conventional computer. In this case, the physical representation of information is dissociated from the course of computation that maps the initial state into the result state. The mapping is formally prescribed and arbitrary with regard to the physical interactions that put it into action. It follows that the physical substrate used to implement the formalism is largely irrelevant in this extreme case. In such a system the state-evolution is contrived by high energy barriers and, often, averaging. As a consequence it is flexible but inefficient with respect to speed and required material. Between these extrema lies a continuum of possible information processing mechanisms that trade-off generality for efficiency.

It is clear that in living systems the representation of information is more closely linked to the physical interaction of the representing structures than in a conventional computer. For example, let us take the case of gene regulation. The representation of the control mechanism is not on the basis of state transitions as it would be the case in a computer program. Instead, the structure of subcomponents is represented and behaviour emerges from the interactions of the subcomponents, i.e., RNA and proteins. The mapping from a DNA sequence to a specific protein structure or a particular RNA secondary structure is largely independent of this structure. The genetic code serves as abstract representation. In case of proteins the separation between processing and representation is facilitated by the essentially arbitrary mapping from codons to amino acids and in case of functional RNA it is provided by the multitude of sequences that can give rise to a particular secondary structure. The behaviour of the control scheme thus represented by DNA, however, is a direct consequence of the physical interactions of the subcomponents.

It has been stated that "the matter that makes up living systems obeys the laws of physics in ways that are expensive to simulate computationally" [13, p. 411]. Conrad offered the conjecture "that it is impossible to simulate [such a biomolecular information processing system] by a machine to which we can communicate algorithms [...] without distorting its rate of operation or the amount of hardware which it requires" [14, p. 227]. The difficulties encountered in attempts to use conventional information technology for the implementation of life-like responses bear out this position.

The need of living systems to process information at a rate sufficient to maintain their material structure ('hardware') within an idiosyncratic set of microstates places special requirements on the substrates suitable to sustain life. This points to a more prominent role of the issue of physical substrates in information processing architectures. Particularly for architectures that ideally would posses life-like features—like, for example, robust real-time behaviour in a complex envrionment, adaptability in paradoxical and ambiguous situations, self-reconfiguration or self-repair—the choice of substrate is likely to be critical.

The physical substrate is certainly also critical to the power of evolutionary processes. It is hardly possible that a process as simple as merely reproduction, variation, and selection could yield systems of sophisticated complexity if not the substrate on which the process acts is amenable to complexification through evolution [15]. The difficulties in demonstrating emergent phenomena in simulated evolution are perhaps due to a problem with the substrate rather than the process [16]. Let us next proceed to an attempt at integrating living matter in a bio-hybrid architecture to, in the long term, endow a robot with some of the capabilities that are not readily accessible to information processing based on a conventional semiconductor substrate.

3 A practical approach

3.1 The information processing of *Physarum polycephalum*

The slime mold *Physarum polycephalum* of the phylum Myxomycota can be found on decaying wood in warm humid forests. Its life-cycle includes a stage in which the organism comprises a single protoplast containing numerous nuclei. This single cell, termed 'plasmodium', moves

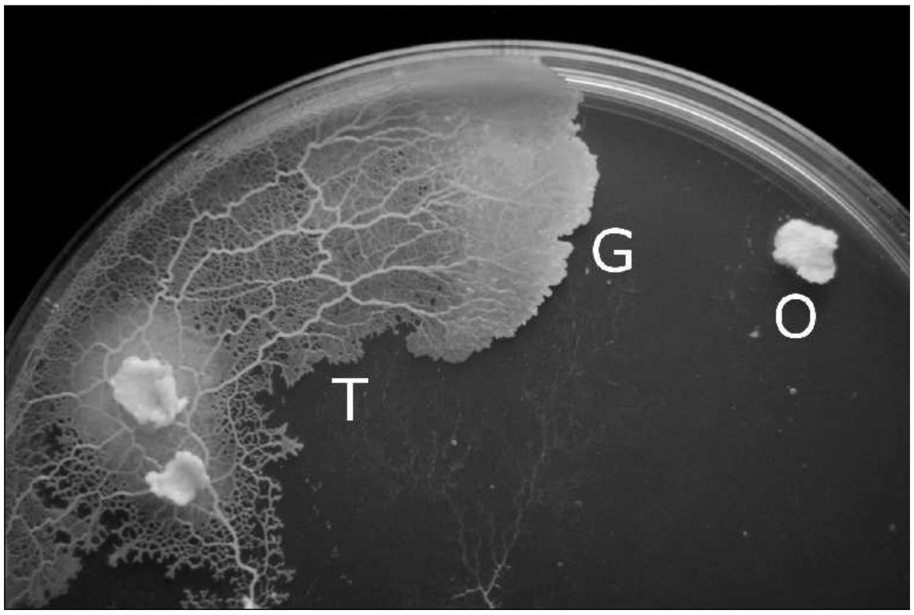


Figure 1: The slime mold *Physarum polycephalum* in the diploid plasmodial state, the most prominent state of its life cycle. The picture shows a section of a Petri dish with 1–2% nutrition-free agar. The plasmodium is a single multi-nuclear cell and distributes material within the cell body through tubular structures (T) which are finely ramified at the growth front (G). Oat flakes (O) are supplied to feed the mold.

in an amoeboid-like fashion and feeds on bacteria and other organic matter. It can easily be grown on a moist agar surface (Fig. 1). Under suitable conditions a plasmodium, which starts out with a few tenth of micrometers diameter, can grow to a giant flat cell exceeding one meter in diameter and harbouring thousands of millions of nuclei.

The behaviour of the plasmodium is size-invariant. The plasmodium acts as a single integrated organism controlled by a decentralised form of information processing. It is found, for instance, that the plasmodium moves towards food sources or away from repellents as a whole cell [17]. Observations have shown too that the plasmodium can find a path through a labyrinth [18].

While in cells of micrometer size signal distribution may be facilitated by diffusion of messenger molecules, the enormous size to which plasmodia can grow necessitates an active communication infrastructure. Being a single cell, this can of course not take the from of a neuronal network. Apparently information is transmitted and processed in plasmodia of *P. polycephalum* by the interaction of local oscillations that also give rise to periodic contractions and expansions of the plasmodium. These spatially synchronised oscillations can be observed in every region of the cell body. If white light, which acts as repellent, shines on a local part of a *P. polycephalum* cell, the oscillation frequency at the stimulated location decreases and desynchronises from the globally synchronised state [19, 20]. The desynchronisation brings about a phase difference between the oscillating rhythm in the stimulated location and oscillations in the remaining parts of the cell. The phase difference propagates to other parts of the cell body through protoplasmic streaming and eventually affects global behaviour and results, for instance, in the escape of the organism from the lit zone.

This form of information processing in plasmodia has been modelled with systems of coupled non-linear oscillators [21, 22]. It also inspired the control scheme for a highly modular robot body with a morphological plasticity that resembles the shape change of a plasmodium [23].

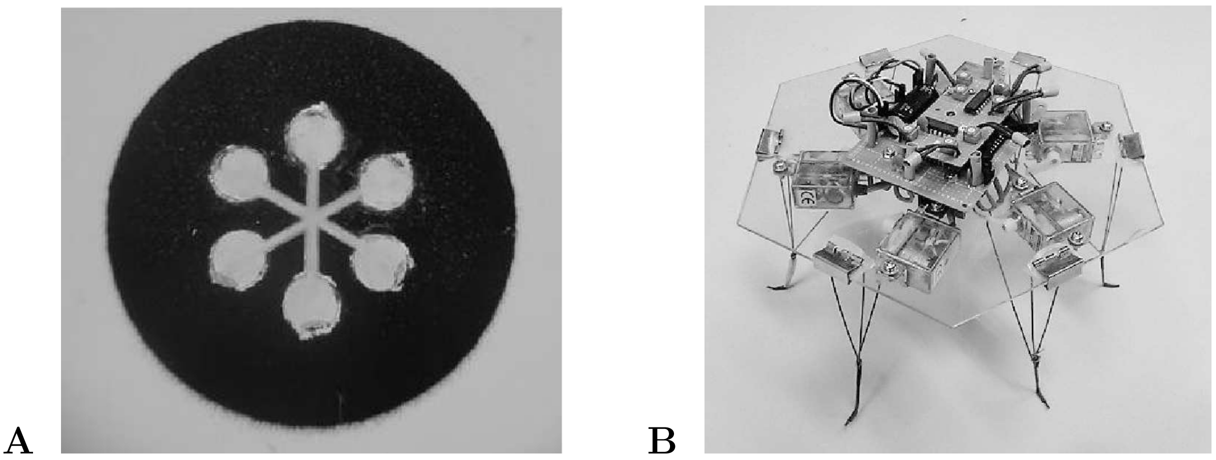


Figure 2: The giant cells of the plasmodial state of *Physarum polycephalum* can be grown into desired shapes through the use of a dry negative mask on a moist surface (A). The six circles at the arms of the pattern have a diameter of 1.5 mm. The part of the plasmodium within each circle can be considered as a non-linear oscillator (cf. [24, 25]). These oscillators are coupled through tubes formed by the plasmodium within the channels. The shape of the cell results in a circuit of coupled oscillators. Living non-linear oscillator circuits patterned in the form shown on the left (A) are used in robot control experiments, where each oscillator controls one leg in a hexapod robot (B). Light signals received by six sensors on the robot are converted into patterns of white-light illuminating the oscillator areas in the physarum circuit.

Our interest here, however, is not in bio-mimicry but in a bio-hybrid system that harnesses the material-based information processing capability of *P. polycephalum* by integrating it into a robot control architecture.

3.2 The bio-hybrid robot system

The bio-hybrid robot consists of three components: a cell, a robot, and an interface connecting the former two. Fig. 2A shows the first component, a plasmodium of *P. polycephalum* grown in a defined two-dimensional shape. We refer to cells configured in such a way as 'physarum circuits'. The omnidirectional six-legged robot driven by this cell is shown in Fig. 2B. The cell is coupled optically to a computer which provides the signal transduction and processing to connect the cell with the robot.

The technique for growing a plasmodium into a predefined physarum circuit makes use of three phenomena. Firstly, *P. polycephalum* requires a moist environment. If it is offered with different surfaces to grow on, it will preferentially grow on a surface from which it can absorb water. Secondly, if a plasmodium of *P. polycephalum* is arbitrarily dissevered by external force, then any fragment large enough to include a nucleus is in general capable to live on. The cut surfaces are swiftly sealed by the gelling of leaking protoplasm and the fragment turns into a small fully functional plasmodium. Thirdly, if two plasmodia come into contact they easily fuse into a single individual organism. The second and third phenomena endow the plasmodia with robustness in an inhomogenous and disruptive natural environment. In combination they also confer a "cut-and-paste" property to the plasmodia which is a great convenience in the making of physarum circuits.

P. polycephalum is cultured in the plasmodial stage on nutrition-free agar at room temperature in the dark and fed with oat flakes (cf. Fig. 1). The shape of the desired physarum circuit is printed with a laser printer on overhead projector foil as clear shape on a solid black background; the background serves to increase contrast in the optical readout described below. In the present experiments the shape depicted in Fig. 2A with the following parameters is used. The six circles have a diameter of 1.5 mm and the width of the channels connecting the circles at the centre is 0.4 mm. The radius on which the six circles are centred is 2.25 mm. The design is based on the idea of Takamatsu et al. [24] but has been modified for the application in robot control and subsequently refined (for earlier versions see [26, 27]). From the printed foil the clear shape is cut away and the foil placed on a 1–2 mm layer or 1.5% agar in a plastic petri dish. Areas in the Petri dish that are covered by the overhead projector foil provide a dry plastic surface and areas in which the agar layer is exposed through cut-outs provide a moist surface. The parameters of the shape that have been chosen for the mask take into account both fabrication and functionality. The area of the plasmodium within a circle is small enough to be regarded as a single oscillator. The channel width is chosen such that the plasmodium will grow a single tube within it, and the length of the channels is a compromise between the speed with which the physarum circuits can be grown and the likelihood that the plasmodium grows over the dry surface of the plastic film.

Next, a plasmodium is grown on the patterned surface. The goal is to have the plasmodium evenly fill the moist area enclosed by the dry, black printed, plastic sheet, but not escape over the dry area or tunnel between the sheet and the agar layer. The growth process is started by filling all six circles in the pattern with small portions from the growth front (labeled G in Fig. 1) of a cultured plasmodium. The Petri dish with the mask is then incubated in the dark at room temperature and about 45–60% relative humidity. The six fragments in the circles of the pattern will first reform into plasmodia. Then they start to grow along the channel towards the centre of the pattern. Upon encountering each other in the centre they fuse and eventual the entire exposed agar surface bounded by the plastic mask is covered with a single plasmodium.

The shape of the plasmodium, visible in Fig. 2A and illustrated in the insert of Fig. 3, results in a system of six coupled non-linear oscillators [24, 25]. For the robot control experiments each oscillator is assigned to one leg of the hexapod which has only a single degree of freedom per leg (Fig. 2B). Thus the oscillation pattern of the cell is transformed into a motion pattern of the actuators and results in locomotion of the robot. Experiments with software oscillators showed that anti-phase oscillation of neighbouring (meta-position) or opposite oscillators (para-position) yields directed motion of the robot. The hexagonal body of the robot carries six light sensors, the signals of which are converted to stimuli applied to the plasmodium, thereby closing the loop between the robot acting in the environment and the *P. polycephalum* cell controlling it.

Another important observation regarding the behaviour of *P. polycephalum* is the basis for interfacing the plasmodium with the robot. Plasmodia of *P. polycephalum* avoid white light, i.e., they show negative phototaxis, but do not respond to orange light near 600 nm [28]. Orange light can therefore be used to follow the thickness oscillations of a plasmodium without disturbing it. A local increase in thickness of the cell is accompanied by locally reduced light transmission. Conversely, white light can be applied to locally stimulate a plasmodium.

Our current experiments focus on the oscillation patterns of a plasmodium with defined shape and the response of the oscillation patterns to local white light stimuli. In these experiments a physarum circuit, prepared as described above, is placed on a orange filtered light table and observed with a camera mounted overhead. For each circle area in the shaped plasmodium the brightness values from a square region of 11×11 camera pixels are averaged. From the spatially averaged values a moving time average with a window length of 15 samples is calculated. Empirical tests showed that for a sample frequency of 0.5 Hz the window length of 15 samples provides the best suppression of camera noise without introducing signal processing artefacts.

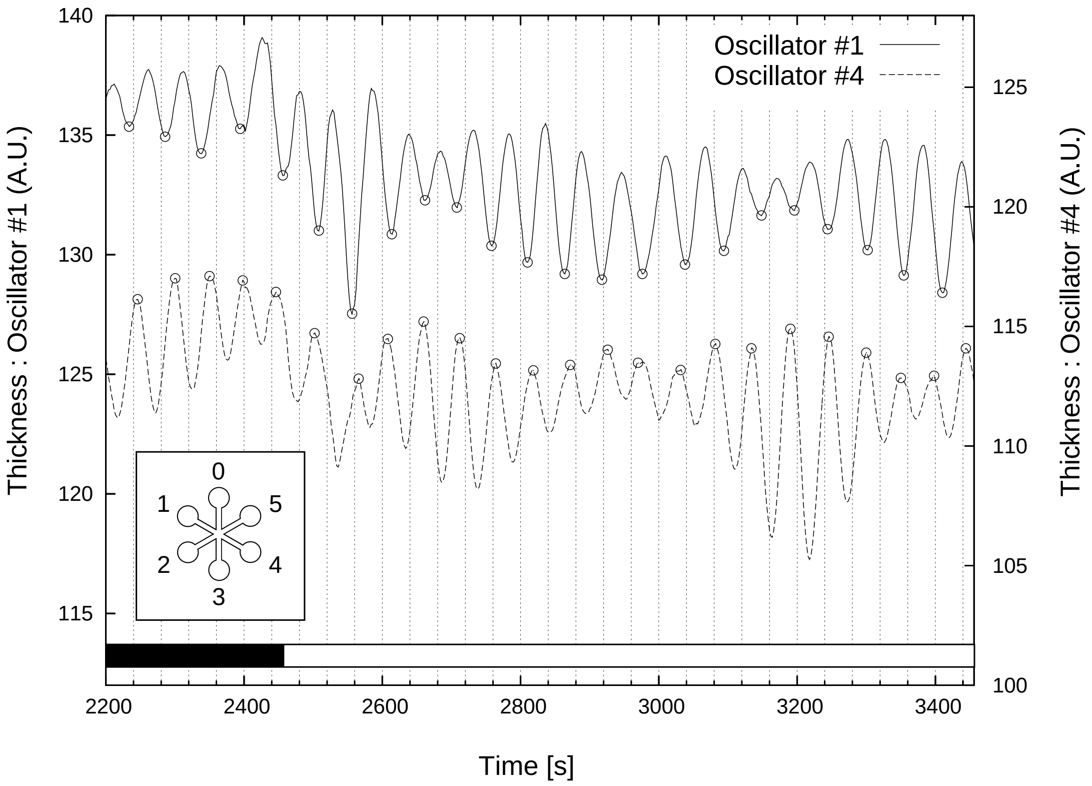


Figure 3: Thickness oscillations in a *P. polycephalum* plasmodium, shown for two oscillators (1 and 4) in para-position. White light applied to oscillators 2,3,5, and 6 starting at t=2452 s stabilised the anti-phase oscillation pattern over an extended period. The insert in the lower left indicates the numbering of the oscillators; thickness scales are in arbitrary units.

Fig. 3 shows results from a typical measurement. The graphs show the variation in thickness determined by light transmission for oscillator 1 and oscillator 4 in a plasmodium shaped as shown in the insert on the lower left. Valleys in the amplitude of oscillator 1 and peaks in the amplitude of oscillator 2 are marked to aid in phase comparison. During the initial period indicated by the black bar near the x-axis the plasmodium spontaneously oscillates. With the two depicted oscillators in near anti-phase a white light stimulus was applied to all oscillators except 1 and 4. The period of the light stimulus is indicated in Fig. 3 by a white bar parallel to the x-axis. The two oscillators fall into anti-phase oscillation which is sustained until t≈3000 s. Short-term cross correlation analysis allows for the detection of anti-phase oscillation among any combination of oscillators. After processing the data to locate the peaks of the amplitudes, the phase relationships among the six oscillators can be determined. The phase difference, ϕ_n, for two oscillators i and j can be calculated from samples (following [29]) as

$$\phi_n = 2\pi\tau_n/T_n$$

where

$$\tau_n = p_n^i - p_n^j \quad \text{and} \quad T_n = p_{n+1}^i - p_n^i,$$

with p_n^i and p_n^j being the n-th peak of oscillator i and j, respectively.

Contractile oscillations in *P. polycephalum* plasmodia that drive the streaming of protoplasm in intracellular tubes (see the region labeled T in Fig. 1) serve as a transport mechanism in the large plasmodia cells. These oscillations are spontaneous and do not require external stimuli.

Having observed spontaneous phase synchronised oscillations, oscillations with 90°-phase shift, and anti-phase oscillations, we found that anti-phase oscillations are common among oscillators in para-position (i.e, located directly opposite to each other).

A number of studies have shown that these oscillations participate in the integration of signals that arrive at different places of the cell body and play a role in the chemotaxis, thermotaxis, and phototaxis of the plasmodia [17, 19, 30–34]. It has been reported that attractive stimuli increase the local oscillation frequency and repulsive stimuli reduce the frequency [17]. The six light sensors of the robot are coupled by local white light stimuli to the plasmodium and accordingly the effect of light stimuli on the global oscillation pattern is of particular interest in this context. Sustained phase relationships map into robot gait patterns and transitions among the synchronisation states of the six-oscillator system give rise to changes of the robot's behaviour. Our investigations are still ongoing, but so far both stabilisation and destabilisation of phase relationships through targeted light input appear practical.

4 Summary

This paper started with life's intrinsic need for information processing which arises from the fact that living systems require active maintenance of their intricate material organisation. Presumably self-organising chemical systems were the precursors of life and their self-assembly properties provided an initial form of dynamic stability that enabled the complexity of some individual systems to rise above what would otherwise be probable [35]. When molecular components then transcended their structural role and became available as information carriers, they did so consistent with their physical interactions. This link between operations carried out on information and the physical interaction among the information carriers appears crucial to the efficiency of biomolecular architectures. However, the abstractions of the current computing paradigm are not well suited to this form of physics-driven information processing and alternatives need to be worked out.

The efficiency in material and energy requirements that comes along with operations on information that are well aligned with the interactions within the physical computing substrate may in particular benefit robotic devices. The threefold challenge of real-time performance, resource limitation, and an unforeseeable complex environment faced by robots is a good proving ground and a likely early application domain for alternative computing substrates. The second part of the paper describes a few humble steps towards the integration of a living cell into a robot control architecture. At this stage the interest is in the fusion of local information into global behaviour through decentralised intracellular processing. In the long-term, such an experimental platform may support the integration and experimental evaluation of concepts from Artifical Life and Synthetic Biology.

Both threads flow together as we argue that a technology capable of mimicking the astounding efficiency of information processing in organisms will need to pay much attention to the physical substrate that enacts the computation. Practical experience with bio-hybrid architectures will be invaluable on the path to extend our present computing paradigm from the formal to the physical.

Acknowledgements

S. T. wishes to thank Moto Kamiura and Shinpei Tatsumi for fruitful discussions. The research reported here was supported in part by the Science and Technology Agency of Japan through the Center of Excellence (COE) program.

References

[1] E. Mendelsohn. *Heat and Life—The Development of the Theory of Animal Heat.* Harvard University Press, Cambridge, MA, 1964.

[2] R. V. L. Hartley. Transmission of information. *Bell System Tech. J.*, 7:535–563, 1928.

[3] A. M. Turing. On computable numbers with an application to the Entscheidungsproblem. In *Proceedings of the London Mathematical Society*, volume 42, pages 230–265, 1937. Corrections, Ibid vol. 43 (1937), pp. 544–546. Reprinted in *The Undecideable*, M. Davis, ed., Raven Press, New York, 1965.

[4] M. L. Minsky. *Computation: Finite and Infinite Machines.* Prentice-Hall, Englewood Cliffs, N.J., 1967.

[5] W. R. Ashby. Some consequences of Bremermann's limit for information-processing systems. In H. L. Oestreicher and D. R. Moore, editors, *Cybernetic Problems in Bionics*, pages 69–76. Gordon and Breach, New York, 1968.

[6] R. Landauer. Irreversibility and heat generation in the computing process. *IBM Journal*, 5(3):183–191, 1961.

[7] R. W. Keyes and R. Landauer. Minimal energy dissipation in logic. *IBM J. Res. & Dev.*, pages 152–157, March 1970.

[8] C. H. Bennet. Notes on the history of reversible computation. *IBM J. Res. & Dev.*, 32(1):16–23, 1988. Reprinted in IBM J. Res. & Dev. 44(1/2):270–277, 2000.

[9] R. Landauer. Fundamental limitations in the computational process. *Berichte der Bunsen-Gesellschaft*, 80(11):1048–1059, 1976.

[10] R. Hanson. Reversible agents: Need robots waste bits to see, talk, and achieve? In D. Matzke, editor, *Proceedings of Workshop on Physics and Computation: PhysComp '92*, pages 284–288, Los Alamitos, 1992. IEEE Computer Society Press.

[11] M. Conrad. *Adaptability.* Plenum Publishing Corp, New York, 1983.

[12] K.-P. Zauner and M. Conrad. Molecular approach to informal computing. *Soft Computing*, 5(1):39–44, 2001.

[13] R. A. Brooks. The relationship between matter and life. *Nature*, 409:409–411, 2001.

[14] M. Conrad. The importance of molecular hierarchy in information preocessing. In C. H. Waddington, editor, *Towards a Theoreritcal Biology*, volume 4, pages 222–228. Edinburgh University Press, 1972.

[15] M. Conrad. Towards high evolvability dynamics. In G. Van de Vijver et al., editors, *Evolutionary Systems*, pages 33–43. Kluwer Academic Publishers, Dordrecht, 1998.

[16] J. F. Miller and K. Downing. Evolution in materio: Looking beyond the silicon box. In *2002 NASA/DoD Conference on Evolvable Hardware (EH'02)*, pages 167–176, July 15 - 18, 2002, Alexandria, Virginia, 2002. IEEE.

[17] A. C. H. Durham and E. B. Ridgway. Control of chemotaxis in Physarum polycephalum. *The Journal of Cell Biology*, 69:218–223, 1976.

[18] T. Nakagaki, H. Yamada, and A. Toth. Intelligence: Maze-solving by an amoeboid organism. *Nature*, 407:470, 2000.

[19] Z. Hejnowicz and K. E. Wohlfarth-Bottermann. Propagated waves induced by gradients of physiological factors within plasmodia of *Physarum polycephalum*. *Planta*, 150:144–152, 1980.

[20] K. Matsumoto, T. Ueda, and Y. Kobatake. Propagation of phase wave in relation to tactic responses by the plasmodium of *Physarum polycephalum*. *Journal of Theoretical Biology*, 122:339–345, 1986.

[21] H. Miura and M. Yano. A model of organization of size invariant positional information in taxis of *Physarum* plasmodium. *Progress of Theoretical Physics*, 100(2):235–251, 1998.

[22] Y. Miyake, S. Tabata, H. Murakami, M. Yano, and H. Shimizu. Environmental-dependent self-organization of positional information field in chemotaxis of *Physarum* plasmodium. *Journal of Theoretical Biology*, 178:341–353, 1996.

[23] A. Ishiguro, M. Shimizu, and T. Kawakatsu. Don't try to control everything!: An emergent morphology control of a modular robot. In *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 981–985, Sendai, Japan, September 28 - October 2 2004.

[24] A. Takamatsu and T. Fujii. Time delay effect in a living coupled oscillator system with the plasmodium of *Physarum polycephalum*. *Physical Review Letters*, 85:2026–2029, 2000.

[25] A. Takamtsu, T. Fujii, and I. Endo. Control of interaction strength in a network of the true slime mold by a microfabricated structure. *BioSystems*, 55:33–38, 2000.

[26] S. Tsuda, K.-P. Zauner, and Y.-P. Gunji. Robot control with biological cells. In *Proceedings of the* Sixth International Workshop on Information Processing in Cells and Tissues*, Aug. 30–Sept. 1, 2005, St. William's College, York*, pages 202–216, 2005.

[27] S. Tsuda, K.-P. Zauner, and Y.-P. Gunji. Robot control: From silicon circuitry to cells. In A. J. Ijspeert, T. Masuzawa, and S. Kusumoto, editors, *BioADIT 2006*, volume 3853 of *Lecture Notes in Computer Science*, pages 20–32. Springer, 2006.

[28] T. Ueda, Y. Mori, T. Nakagaki, and Y. Kobatake. Action spectra for superoxide generation and UV and visible light photoavoidance in plasmodia of *Physarum polycephalum*. *Photochemistry and Photobiology*, 48:705–709, 1988.

[29] S. Nakata, T. Miyata, N. Ojima, and K. Yoshikawa. Self-synchronization in coupled salt-water oscillators. *Physica D*, 115:313–320, 1998.

[30] W. Korohoda, L. Rakoczy, and T. Walczak. On the control mechanism of protoplasmic streamings in the plasmodia of *Myxomycetes*. *Acta Protozoologica*, VII(29):363–373, 1970.

[31] T. Ueda, K. Matsumoto, and Y. Kobatake. Spatial and temporal organization of intracellular adenine nucleotides and cyclic nucleotides in relation to rhythmic motility in *Physarum* plasmodium. *Experimental Cell Research*, 162(2):486–494, 1986.

[32] T. Ueda. Self-organization of *Physarum polycephalum*: Phase locking and information control. In K. Toko and G. Matsumoto, editors, *Self-Organization*, chapter 4, pages 86–102. Asakura Shoten. Japan, 1996. In Japanese.

[33] T. Nakagaki, H. Yamada, and T. Ueda. Modulation of cellular rhythm and photoavoidance by oscillatory irradiation in the *Physarum* plasmodium. *Biophysical Chemistry*, 82:23–28, 1999.

[34] J. Tanaka and Y. Miyake. Modulation of intracellular rhythm and behavior of *Physarum* plasmodium under the condition of mutual entrainment. In *Proceedings of the SICE 2000 Conference*, pages 212–A4. The Society of Instrument and Control Engineers, 2000.

[35] J.-M. Lehn. Toward complex matter: Supramolecular chemistry and self-organization. *PNAS*, 99(8):4763–4768, 2002.

Effect of Multi-Level Fitnesses on the Evolution of Multicellularity in Artificial Organisms

Moritz Buck and Chrystopher L. Nehaniv

Adaptive Systems Research Group
School of Computer Science
University of Hertfordshire
College Lane
Hatfield Herts AL10 9AB
U.K.

Abstract

We study how groups of genetically identical agents evolve under the constraints of a fitness function representing two different levels of selection. At a high level a group of agents needs to create a defined pattern but agents are also rewarded for a lower level behaviour which is much easier to sustain and competes with the high level one.

A grid of artificial cells model a population of closely related cells, all having the same Artificial Genetic Regulatory Networks (GRNs) controlling them. The GRNs are coded into a bit string genome and populations of these are evolved using a Genetic Algorithm.

Previous work on the transition between single cell organisms and multicellularity leaves open a lot of questions on how this big step in evolution happened. One of those questions is about the transition of fitness at a low level of organization to a fitness at a higher level. With our model we study this question and try to see what insight we can obtain with artificial organisms.

1 Cooperating Cells

August Weissman devised during the late nineteenth-century a theory which is still accepted nowadays, postulating early segregation of somatic and reproductive cells in multicellular organisms. Even if it was developed on the false hypotheses that this was instigated by the loss of the hereditary information of the somatic cells, it clarifies how selection can happen in multicellular organisms at a higher level, why the individual is the whole and not the cells or the genes. But this theory, as Buss describes in [2], cannot explain the transition from one level of organization to the other.

Evolution probably started with simple chemical replicators, which then somehow teamed together to build the first primitive regulation networks, those groups of molecules then chose to hide themselves behind a wall, so that they wouldn't be troubled that much by a hazardous environment: the first proto-cell. These simple cells "ruled" over the living world for hundreds of millions of years and still do with their modern day cousins, bacteria, which are still the most represented genre.

Before the advent of differentiated multicellular organisms around the time of the Cambrian explosion, for billions of years, clones of microbes already had been living together (e.g. in the Precambrian sediments colonized by microbial mats [5]). Nearby cells of the same species were likely to be closely related, simply due to spatial constraints on reproduction, with descendants of a single cell living in close proximity or direct contact with each other. Due to inclusive fitness pressure, it is more than likely that such clones of closely related cells would cooperate to some extent. Such a stage of *multicellularity via aggregation* eventually lead to *differentiated multicellularity* with cells differentiating into different cell types in a cooperative division of labour. The origin and persistence of such higher level organization in advent of differentiated multicellular *individuals* is still a little understood major transition in evolution [2, 6, 7].

Little is known about the constraints and the requirements for this transition in behavior. Michod [7] describes how the mutation rate can influence the stability of a multicellular organism against defection from constituent cells, and the theory of Weissmann suggests how the higher level individual can fight against freedom of a lower level through segregation of the germ and somatic cell lines.

In this article an artificial life model based on interacting agents will be used to investigate the influence of different factors on the evolution of multi-agent (cellular) systems. Such systems will undergo an artificial evolution, with two-level fitness functions to emulate the two levels of selection of early cooperating cells, and study under what kind of conditions artificial multicellular agents can be evolved.

2 Methodology

To explore ways to build multicellular artificial life models, one needs different ingredients: first, a single cell and some way to model its behavior. The cells in this experiment will be controlled by bio-inspired Genetic Regulatory Networks [1, 8, 10, 12]. Then those single cells need to be arranged in a certain way, here a 2D grid with a 4-cell von Neumann neighborhood, similar to the ones used for Conway's game of life [3], to present the possibility of interaction with genetically identical neighbors. And we need to have a way to design the control system. We will evolve the GRNs using a Genetic Algorithm. In the series of experiments an aggregation of cells — potentially comprising a single higher level individual — will be modeled by agents arranged in a cellular automaton (CA) type of grid. Genetic Regulatory Networks (GRNs) will control these agents, the GRNs being constructed in the course of evolution with a Genetic Algorithm (GA).

2.1 GRN Model

A variant of the GRN model described in [4] is used which was based on the earlier one in [9]. Genetic regulatory networks, as their name imply, are inspired by the regulation systems found in biological cells to regulate gene expression [11]. In a simplified model of biology, the genetic information which encodes for proteins, which are the building blocks of the cellular machinery, is separated into genes. A gene is a unit which produces one protein, the production of each gene can be controlled by certain other proteins which attach themselves to certain regions of the gene, called regulatory cis-sites. Proteins can reduce or increase the production of genes by binding to their cis-sites. Some cis-sites also require the binding of more than one specific protein to be able to regulate the production of a protein by the gene.

In the model used here a gene will produce a varying amount of a certain protein, each gene will be able to have any number of cis-sites and each cis-site can have any number of binding sites. The cis-sites of one gene have an additional synergetic property, the activation levels of all the

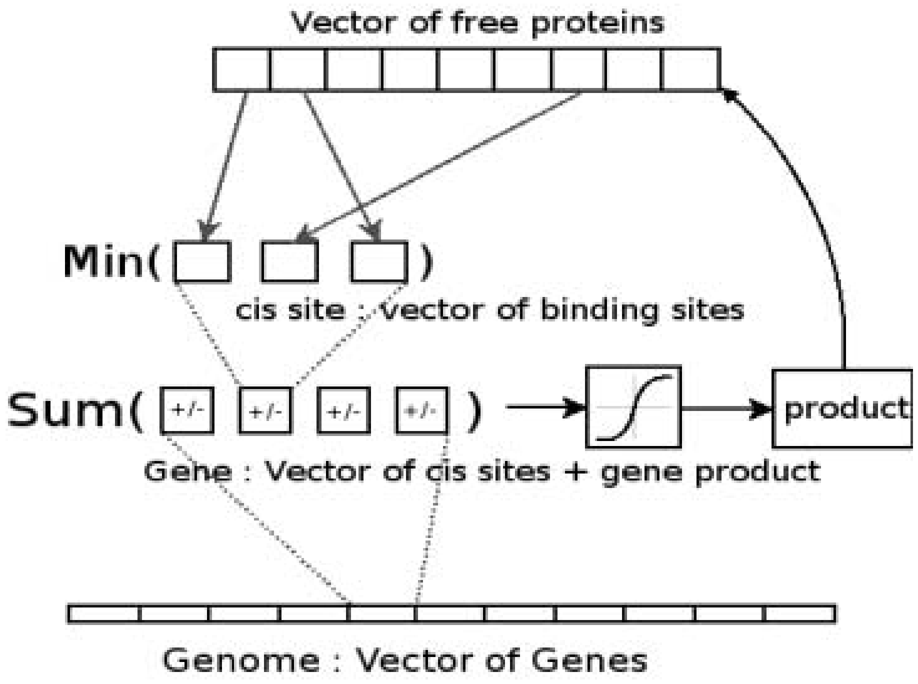


Figure 1: Schematic representation of an artificial genetic regulatory network

cis-sites are summed to determine the amount of the gene's protein produced. In this model the different binding sites of one cis-site will have an exclusive effect, with the (inhibitory/activatory) activation of the whole cis-site equaling the activation created by the binding site which has the lowest amount of protein bound.

The whole network is represented by any number of such genes, producing proteins from a finite set. All such genes are encoded in a bit string along with some other information necessary for the model. Details follow.

2.1.1 GRN Mathematics

Each gene from the network will produce during one time step of a simulation a certain amount of a protein. This amount of protein will be computed from the sum of the activation levels of the cis-sites of this gene. The activations of each cis-site are computed as the minimum of the activations over all the binding sites of this cis-site, and the activations of each binding site are computed by adding the amount of free protein of the protein to be bound to that site and the already bound proteins on this site. We get the bound amount $b^t_{i,j,k}$ at time step t for a certain protein z binding to the binding site k of the cis-site j of the gene i binding the expression:

$$b^t_{i,j,k} = \beta f^t_z + \tau b^{t-1}_{i,j,k},$$

where τ is a parameter describing the protein decay rate, β a parameter describing the binding proportion of free type z protein, $b^{t-1}_{i,j,k}$ the bound amount at the previous time step, which can be seen as the amount of protein still bound and f^t_z the amount of free protein z available for this binding site: $f^t_z = p^t_z/n_z$, with p^t_z the free amount of protein z and n_z the number of sites binding protein z all over the genome. We can therefore see the activation of a binding site $b^t_{i,j,k}$ as the amount of the protein bound to that site.

The total activation $c^t_{i,j}$ of the cis-site j is the minimum activation over all its binding sites:

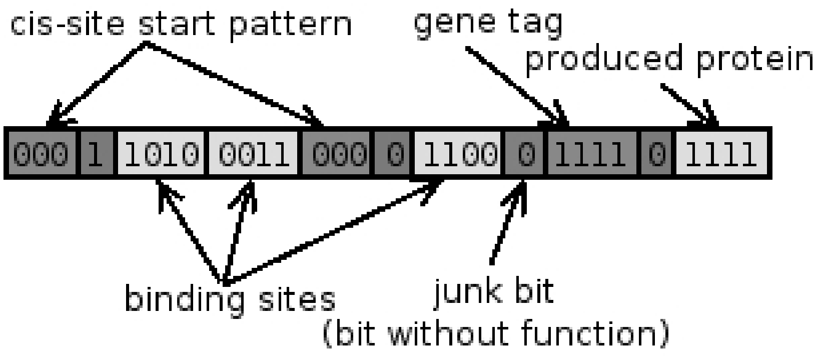


Figure 2: The Structure of one Gene

$$c_{i,j}^t = \min_{\text{all } k} b_{i,j,k}^t,$$

the total activation a_i^t of gene i at time t becomes:

$$a_i^t = \sum_{\text{all } j} \pm c_{i,j}^t,$$

where the sign depends on the type of cis-site, either *activator*, then the sign will be positive or *repressor*, negative. Once the activation has been computed, the production is computed using one of two sigmoidal transformation, one for a gene which is "on" by default and one "off". Thus gene i will produce P_i^t proteins where

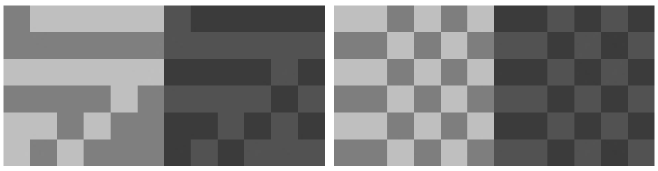
$$P_i^t = \frac{r}{2}(\tanh(\frac{a_i^t + o}{s}) + 1),$$

where o is -15 if this gene is "off"by default and $+5$ if "on", r and s are range and steepness parameters (here $r = 150$ and $s = 5$ as in [4]). To conclude the computation the amount of protein which has been bound at that time step is removed from the free amount of protein and the remaining amount of free protein is multiplied by the decay parameter τ, the freshly produced proteins get added to the free proteins and if the total amount of a certain protein exceeds a saturation parameter it will be set equal to this parameter. In this article, 16 different protein are used which can each be produced by multiple genes and bind to their specific binding sites (possibly a single protein may bind to many different cis-sites within a gene or between genes). In addition to possible regulatory functions as already described, certain of those proteins will have some specific role in the behaviors of the agents and the evolutionary algorithm as we will see in later sections.

2.1.2 Encoding the Network

This genetic regulatory network of a cell is represented by a binary string. The first bits of the genome represent some parameters of the model (11 bits here): τ the decay rate, β the binding proportion, and the saturation value. The values of those parameters are mapped via a lookup table (see [4] for more details) based on the binary representation.

The rest of the genome encodes the network itself. The different functional regions of the genome are tagged with 2 short binary patterns, one pattern coding for the start of a cis-site and one pattern signaling the start of a gene. The tag for a gene (pattern: '1111') precedes a



(a) lattice early during the run

(b) lattice later in the run

Figure 3: Image of the cell lattice. Left in each pair (a & b): the statuses of the cells, here either the cells are either in "red" (dark) or "green" (light) state and the whole multicellular organism is rewarded for building a checkerboard-pattern. Right in each pair (a & b): cells which are "communicating" with their neighbors are colored lighter.

bit controlling whether the gene is on or off. This precedes a 4 bit binary string encoding the protein itself ('0000' for protein '0', '0001' for protein '1', ... , '1111' for protein '15'). The region before the gene tag and after the last protein delimiter is the regulatory system for that gene, which can be composed of numerous cis-sites. Each cis-site is started by the cis-site tag (pattern: '000') and then followed by a bit coding if the site is an activating or a repressing one. Each cis-site tag is then followed by binding sites until the next cis-site tag. Each of those binding sites is a 4-bit pattern encoding for the protein it binds, similar to the patterns used to describe the produced protein. Each bit of the genome can only have one function so the determination of the functions follows a strict order and set of rules so that there is no overlapping and some patterns encoding for multiple functions can be discriminated.

2.2 Agents and Multicellular Array

The agents in the modeled multicellular array represent a colony of artificial cells, or a primitive multicellular organism. The cells are guided by two different goals (fitness functions), one at a cellular and one at the multicellular (or organism) level. The multicellular goal is to build a certain pattern for which the whole will be rewarded, whereas the single celled individual would be rewarded for staying quietly in a certain state. The multicellular array is a 2D array with toroidal borders. Each field of the array is an agent (cell) which will be controlled by a GRN. In these experiments the GRNs controlling all the agents in a given array will be the same. The neighborhood is a four agent neighborhood. The update of the cells is randomly asynchronous, with all the cells in one time step updated once in a random order.

The agents can interact with each other using a set of proteins, 4 proteins used to define the cells with which the cell will communicate. Each of those proteins, depending on their level, either opens or closes a communication link to one of the neighbor cells. If one of those links is open a proportion of an other protein will diffuse to the neighbors to which links are open. Each cell can be in one of three different "states". Two "multicellular" states "green" and "red" to set up a pattern and one "individualistic" state. The level of one specific protein defines whether the cell is "individualistic" or not, if it is not, a second protein level determines in which of the two "color" states of the desired pattern the cell is.

2.3 Genetic Algorithm

A genetic algorithm is used to model the evolution of the agents. We evolve 125 genomes for 500 generations, with weak elitism (the overall best is kept once in each generation). The implementation of the mutation is straight forward: each bit of the genome has a fixed probability

(0.001%) to mutate. A two-point cross-over is used, where the points of the cross-over can only be at the level of either a cis-site start or at a gene tag, so that the functional blocks are not "cut". The length of the transmitted material need not be the same for each of the parents, so the length of the genomes varies.

2.3.1 Fitness Functions

The fitness function is separated in two different constituent ones: one high level fitness function and a low level one, the high level one needing cooperation between the cells.

The low level fitness is, for the duration of a simulation, the maximum time a cell has been in the "individualistic" state, divided by the length of the simulation :

$$f_l = \max_{\text{all cells}} (\sum_t s_i^t)/t_{\max}$$

where s_i^t is equal to 1 if the cell is in "individualistic" state at time step t and otherwise 0, and $t_{\max}$ the length of the simulation. The higher level fitness is depending on the higher level pattern desired, at present only one pattern has been implemented in the model, the checkerboard pattern. Each cell c of the lattice gets a score depending on the states of its neighbors: if it is not in the "individualistic" state

$$s_c = 1/2 + \sum_{i=0}^{3} \begin{cases} +\frac{1}{8}, & \text{if the } i^{\text{th}} \text{ neighbor is in a different state but not "individualistic"} \\ -\frac{1}{8}, & \text{else,} \end{cases}$$

otherwise $s_c = 0$. The mean of this score is then taken over the simulation time and the number of cells:

$$f_h = \frac{1}{t_{\max}\, d^2} \sum_{\text{all cells } c} \sum_t s_c,$$

where $t_{\max}$ is the length of the simulation and d^2 the number of cells (d the number of cells on one side of the square environment). Both fitness scores range between 0 and 1, 1 being the best fitness. The final fitness for one multicellular entity's lifetime will be the maximum of those two fitnesses with a weight $\alpha \in [0, 1]$ applied on f_l :

$$f = \max(\alpha f_l, f_h)$$

This α parameter can be described as a defection parameter, the higher α the easier it will be for the single cells to be egoistic and not add anything to the goal of the cooperating colony. The individualistic state being much easier to reach, the α parameter sets, in an optimization formulation, an easy to reach local fitness maximum. The idea is to vary α to see how easily can the population quit this sink.

The update being in a random order, to get the fitness of one individual in the genetic algorithm we do 5 simulations for each genome and average the fitnesses.

Selection is done through tournament selection, 25 multicellular individuals are picked in the population of genomes and the 2 best ones can reproduce to the next generation, with cross-over in 30% of the cases.

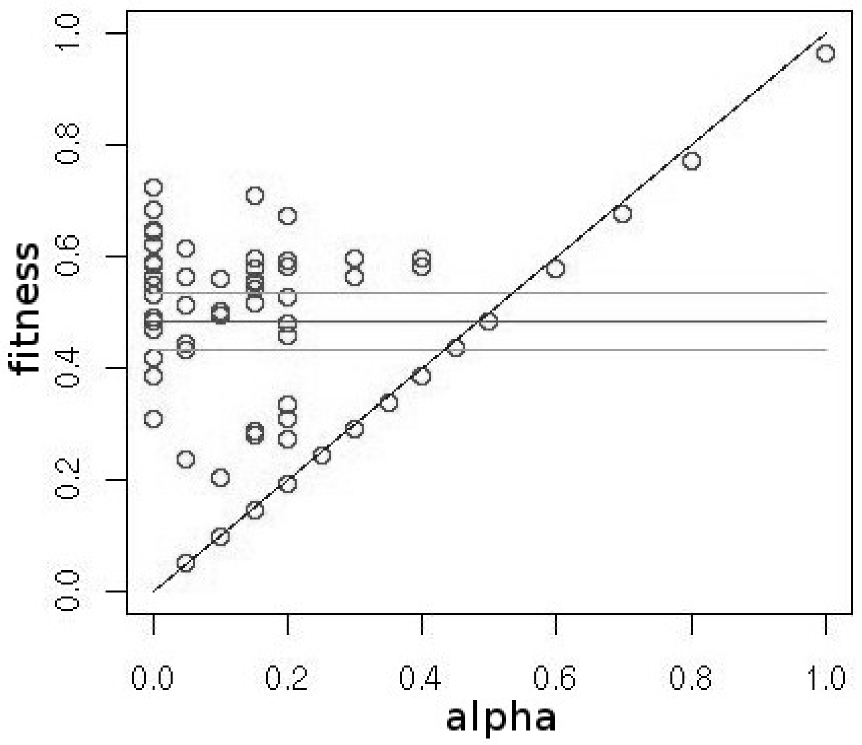


Figure 4: Fitness vs. α, α is the weighting on lower level "individualistic" behaviour. Circles: results of evolutionary runs, diagonal line: line of equation $f = \alpha$, dark horizontal line and lighter ones: mean fitness of random population and zone of 95% confidence.

2.4 Experimental Setup

In a first series of experiments, the target is to see if this model is able to evolve cooperating agents at all. For this setup we will only use the fitness function f with $\alpha = 0$ so $f = f_h$, the high level fitness function. A fitness close to 1 indicates ability to set up the states of the cells in a checkerboard manner (fig. 3). We carry out 10 runs of the genetic algorithm, with population size 125 for 500 generations, and study the dynamics of the best individuals. To complete this study we generated 10 times 500 times 125 random genomes and computed their fitnesses, to get a control experiment.

The second experimental setup is designed to study the effect of the α parameter. The genetic algorithm is run several times for different values of α, and the fitnesses (colony and individual) of the best element of each evolutionary run are studied.

3 Results

For the first set of experiments, the genetic algorithm is able to design GRNs adopting the desired checkerboard-pattern at some level. The best fitnesses obtained for $\alpha = 0$ is around 0.7. Most of the good GRNs chose to use the communication proteins only in the first steps of the simulation, to set the pattern up and then go on on their own, and only communicate again if some perturbation occurs. They use the differences in protein levels created in the first few rounds (created solely by the random update which is the only non-deterministic part of the model) to choose their state. In the best GRN (fig. 3) only one type of cell communicates, the red ones, so if a cell has a high level of the communication protein it is probably surrounded by red cells so it is switched to the other cooperative state (the green one). Some organisms showed periodic or pseudo-periodic behaviors, going into one checkerboard pattern and then switching

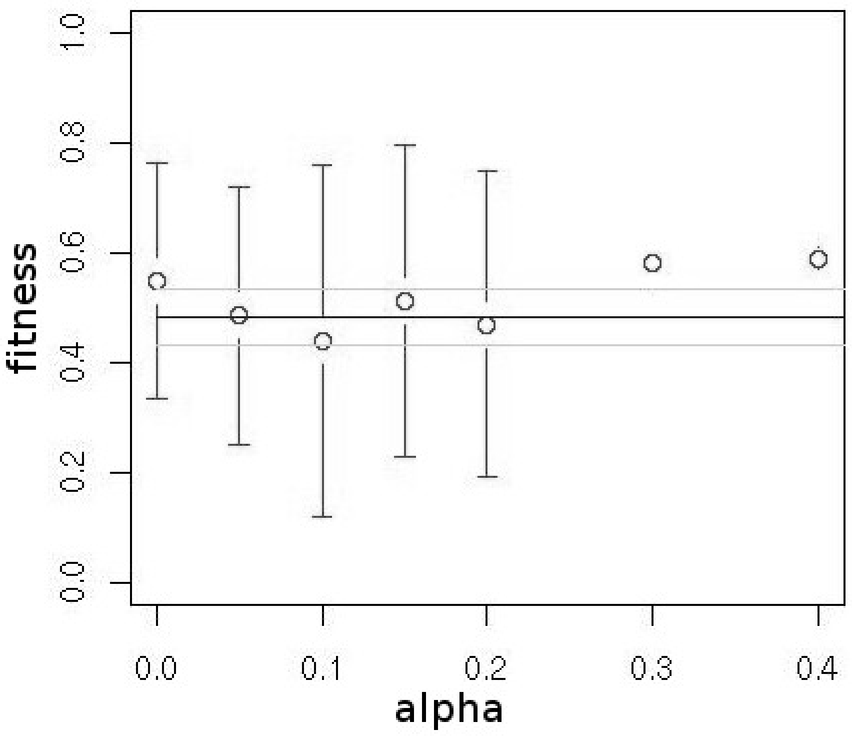


Figure 5: Fitness vs. α for those simulations which achieved multicellularity. Circles: mean fitness of all experiments with a given α having achieved multicellularity, bars: 95% confidence interval; dark horizontal line and lighter lines: mean fitness of random population and zone of 95% confidence.

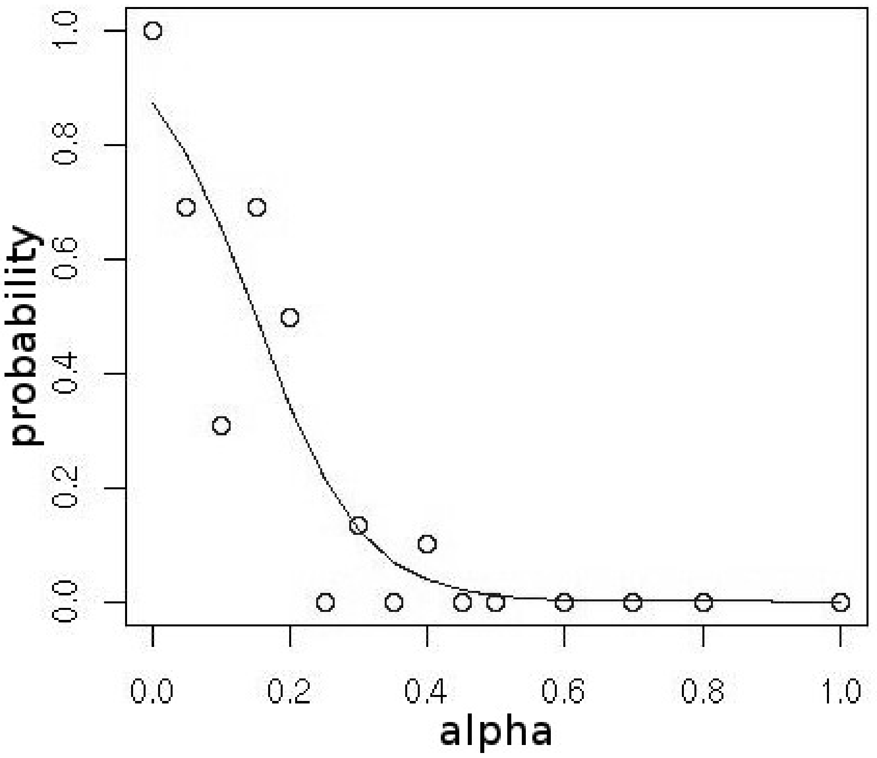


Figure 6: Frequency of achieving multicellularity vs. α. Circles: Measured proportion of evolutionary simulations which did converge to a multicellular state for a given level α of individualistic weighting, line: estimation of the probability function for achieving the transition to multicellularity.

between red and green cell states synchronously.

A noteworthy remark would be that the control experiment did not perform very badly (fig. 4, fig. 5, horizontal lines). It achieved fitnesses on the average of the same level as the evolved GRNs. The difference being that the evolved GRNs have a higher variability for their fitness, hence the evolved GRNs are able to achieve many results which the random ones where not able to.

In the second experiment the effect of α is studied. The first remark is that the maximum for the lower level fitness is very easy to reach (the maximum for that fitness is not exactly α because in the first time step of the simulation the cells will never be able to be in the "individualistic" state), in fig. 4 we see that for α above 0.4 every evolutionary run actually converges to an "individualistic" state and hence cannot achieve multicellularity, and for even the smallest values for α above 0 some runs stay stuck in this non-cooperative state. But once the population manages to get out of the "individualistic" state, it will achieve a level of multicellularity independent of α. In fig. 5, only the simulations achieving multicellularity have been plotted, and independently from α, all achieve levels of fitness of the same order. The probability of achieving multicellularity, plotted in fig. 6, shows a very steep decrease for values between 0 and 0.4, this decrease is the due to the competition between the two levels of selection, the "individualistic" state being much easier to achieve, it very quickly overtakes the cooperative state. But this kind of sharp decrease looks similar to a phase transition, so we suppose that if we increase the population size we will have an even more steep decrease, with a cut-off value under which every simulation reaches multicellularity and above which none achieves it.

4 Discussion

It seems that the evolution of interacting agents using GRNs for some simple problems is possible. These colonies of interacting agents can be seen as very simple "multicellular" artificial organisms. These organisms were able to evolve under the influence of multiple fitness function which select on different levels of behaviour by the organisms. The fact that multicellular organisms can be sustained in a environment with selection on multiple-levels has been studied by Michod in [7], where he used dynamical systems analyses to show that, supposing multicellularity has been able to evolve, the multicellular cooperation in colonies can survive competition with selfish individuals which arise through mutation, for mutation rates below a threshold. In our agent-oriented approach we could actually evolve simple multicellular colonies which had to compete with some lower level constituent cells, depending on a parameter weighting the influence of those lower level foes. But we still are limited to the artifice of genetic algorithms which imposes an explicit fitness function, the next step would be to develop a model with a more "natural" evolution, with an implicit fitness built into the model itself. For example introducing mutations and reproduction inside of the multicellular grid would permit the study of the problem of the arising of multicellularity in a more credible and useful context.

With this work we have shown that it is possible to evolve, with classical genetic algorithm methods, simple multicellular systems controlled with GRNs. And perhaps more importantly it shows that we can use this kind of very simple system as a platform to study different transitions in evolution such as multicellularization, differentiation or social behavior in artificial systems. This kind of approach might not have a direct impact on the understanding of biological evolution, but it could help to formulate certain hypotheses to be tested on real living systems, by giving insight into possible mechanisms, or, alternatively, from another perspective it could help to develop massively parallel computing devices in a long term view.

References

[1] Wolfgang Banzhaf. On the dynamics of an artificial regulatory network. In *Advances in Artificial Life (Proc. European Conference on Artifical Life - ECAL'03)*, volume 2801 of *Lecture Notes in Artificial Intelligence*. Springer Verlag, 2003.

[2] Leo W. Buss. *The Evolution of Individuality.* Princeton University Press, 1987.

[3] Martin Gardner. Mathematical games: The fantastic combinations of John Conway's new solitaire game "life". *Scientific American*, October 1970.

[4] Johannes F. Knabe, Chrystopher L. Nehaniv, Maria J. Schilstra, and Tom Quick. Evolving biological clocks using genetic regulatory networks. In *Proceedings of the Artificial Life 10 Conference (Alife X).* MIT Press, 2006 (in press).

[5] Lynn Margulis. *Early Life.* Science Books International, 1982.

[6] John Maynard Smith and Eőrs Szathmáry. *The Major Transitions in Evolution.* Oxford University Press, 1995.

[7] Richard E. Michod. *Darwinian Dynamics: Evolutionary Transitions in Fitness and Individuality.* Princeton University Press, 1999.

[8] Chrystopher L. Nehaniv. Self-replication, evolvability and asynchronicity in stochastic worlds. In *Proc. 3rd Intl. Symp. on Stochastic Algorithms, Foundations and Applicaitons (SAGA 2005)*, volume 3777 of *Lecture Notes in Computer Sciences*, pages 129–169. Springer Verlag, 2005.

[9] Tom Quick, Chrystopher L. Nehaniv, Kerstin Dautenhahn, and Graham Roberts. Evolving embodied genetic regulatory network-driven control systems. In *Advances in Artificial Life (Proc. European Conference on Artifical Life - ECAL'03)*, volume 2801 of *Lecture Notes in Artificial Intelligence*, pages 266–277. Springer Verlag, 2003.

[10] Torsten Reil. Dynamics of gene expression in an artificial genome - implications for biological and artificial ontogeny. In *Advances in Artificial Life (Proc. European Conference on Artifical Life - ECAL'99)*, volume 1674 of *Lecture Notes in Artificial Intelligence*, pages 457–466. Springer Verlag, 1999.

[11] Maria J. Schilstra and Hamid Bolouri. The logic of gene regulation. Abstract for the 3rd International Conference on Systems Biology, December 11-15, 2002, Karolinska Institutet, Stockholm, Sweden, 2002.

[12] Jiri Vohradsky. Neural network model of gene expression. *The FASEB Journal*, 15:846–854, 2001.

A Rule-Based Approach to Selecting Developmental Processes

Jan T. Kim, jtk@cmp.uea.ac.uk
School of Computing Sciences, University of East Anglia
Norwich NR4 7TJ, United Kingdom

Abstract

This paper introduces a method to generate synthetic regulatory gene networks (RGNs) using a predetermined developmental process, specified by a computer model. The candidate RGN is implanted into the developing system and a growth record comprised of gene expression levels in the system's component and the developmental steps affecting each component is obtained. Performance of an RGN is scored with an objective function based on correlations of gene expression levels and execution of developmental steps. Optimisation of the objective function yields synthetic RGNs that generate gene expression patterns which can be used to control the desired developmental process.

The concept is demonstrated using an elementary model of plant growth to perform optimisation of the numerical parameters of randomly generated synthetic RGNs, but it can be generalised to any rule based computational model of development.

1 Introduction

The complexity of development and evolution of developmental processes has intrigued bioscientists for centuries. This focus of interest has been inherited by Artificial Life, where numerous computational models and frameworks to simulate and study development and its evolution have been published [Wilson, 1989, Fleischer and Barr, 1993, Kim, 2000, Bentley and Clack, 2005].

The role of regulatory gene networks (RGNs) for organising differentiation and development based on genetic information has been recognised already by [Kauffman, 1987]. Recently, computational simulations of RGNs have attracted much renewed interest, due to progress in Molecular Systems Biology and the emergence of "Network Biology" [Barabási and Oltvai, 2004] as part of a new trend of network based methods which begins to unify research in various fields including physics, social sciences and biological disciplines. Network approaches are applied on multiple levels of biological organisation, where RGNs and ecosystems [Williams and Martinez, 2000] are perhaps the most prominent ones.

For Systems Biology, computational simulation of gene expression dynamics is important as a basis for developing and objectively evaluating methods for analysing gene expression data. Among the simulators used for researching methods for gene expression data analysis and RGN reconstruction are Gepasi [Mendes et al., 2003], `transsys` [Repsilber and Kim, 2003, Kim, 2005], and the SynTReN simulator which has recently been developed specifically for this purpose [Van den Bulcke et al., 2006].

Generating synthetic gene expression data requires biologically realistic RGN models. Various methods for using empirical measurements of gene expression data to reconstruct RGNs have been proposed and developed [Rung et al., 2002, Basso et al., 2005]. However, using such reconstructed RGN models for evaluating gene expression analysis methods leads to a problem of circularity: The reconstruction procedure involves the methods to be analysed, and generation of the test data is thus not independent of the system to be tested.

In addition to models directly derived from empirical data ("life as it is"), the scientific object of Artificial Life includes systems that are based on the same principles as real biological systems ("life as it could be"). Computational systems of the latter type are not dependent on specific empirical data, and synthetic data generated from such computational models can therefore be used to assess RGN reconstruction methods in a way that is not affected by the circularity described above.

This report introduces an approach for generating and parameterising synthetic, computational RGN models that are capable of organising a formalised process of biological development. The computational model used to demonstrate this approach is `L-transsys` [Kim, 2001], a system for modelling plant morphogenesis based on a combination of Lindenmayer systems (L-systems) and `transsys`. RGN models that are suitable to control a predetermined process of plant development are generated by parameter fitting.

This concept requires a method to measure the similarity between the predetermined developmental process and the actual process obtained with a candidate RGN. Assessing similarity of phenotypes, or of developmental processes, is generally a difficult problem. The choice of phenotypic traits to use for comparison can be subjective, and similarity measures of 2- or 3-dimensional objects are frequently computationally expensive. The method presented here avoids these problems by focusing on the spatiotemporal sequence of developmental steps. An elementary developmental step in a computational model of development can generally be formalised as the execution of a growth rule [Kniemeyer et al., 2004]. The sequence of rule activations during a developmental process is a measurable trait that is computationally readily accessible and not dependent on subjective choice.

As an initial approach towards this concept, this paper explores the use of parameter optimisation to produce RGNs that can organise the growth of a branching structure (a classic motif in plant modelling with L-systems). The pattern of rule activation in the growth process is used to develop objective functions, and the RGN parameters are fitted to optimise these objective functions using a local search procedure.

2 Methods

2.1 Background

`transsys` [Kim, 2001] is a computational system for modelling RGNs. A `transsys` program consists of declarations of the factors (proteins or other gene products) and the genes which constitute the RGN. Fig. 1 shows an example of a `transsys` program. A `transsys` instance contains the concentration values of all factors in the program and thus describes the state of a biological unit (such as a cell or a plant component in the case of `L-transsys`).

Lindenmayer systems (L-systems) [Prusinkiewicz and Lindenmayer, 1990] represent the morphology of a plant by a string of symbols. In `L-transsys`, a symbol may be parameterised by a `transsys` instance. Plant growth is simulated by applying a set of substring replacement rules to the string representing the current plant morphology. These replacement rules represent the elementary steps of development.

```
transsys example
{
  factor A
  {
    decay: 0.1;
    diffusibility: 0.2;
  }

  factor R
  {
    decay: 0.15;
    diffusibility: 0.3;
  }

  gene agene
  {
    promoter
    {
      constitutive: 0.01;
      A: activate(0.02, 1.0);
      R: repress(0.1, 1.1);
    }
    product
    {
      default: A;
    }
  }

  gene rgene
  {
    promoter
    {
      A: activate(1.0, 10.0);
      R: repress(1.0, 1.0);
    }
    product
    {
      default: R;
    }
  }
}
```

Figure 1: An example `transsys` program consisting of two factors and two genes. The numerical parameters in this program are 0.1, 0.2 **(factor `A`)** 0.15, 0.3 **(factor `B`),** 0.01, 0.02, 1.0, 0.1, 1.1 **(gene `agene`) and** 1.0, 10.0, 1.0, 1.0 **(gene `rgene`).**

`L-transsys` is combines the `transsys` and L-systems by associating `transsys` instances with L-systems string symbols. The current `transsys` software archive, which provides more detailed technical documentation, is available from the `transsys` home page.[1]

Fig. 2 shows the `L-transsys` program of a simple branching structure. The `transsys` program `aux` provides one factor, `A`, and a gene `agene` which constitutively expresses one unit of `A` per time step. As `A` does not decay, it accumulates at a rate of one unit per time step. This factor is used to trigger execution of the rule `grow` every 5 time steps. The rule `grow` states that a `meristem` symbol in which contains at least 5 units of factor `A` is replaced by the sequence of symbols to the right of the arrow, the righthand side (RHS). The statement `transsys t: A = 0.0`, attached in parentheses to the `meristem` symbols in the RHS, specifies that the `transsys` instance of the original `meristem` symbol is copied, and the amount of factor `A` is set to 0 in the process. Fig. 3 shows the development of this simple branching structure.

The `transsys` program `aux` serves as a host into which other programs are implanted in the process of parameter optimisation (see below). Therefore, the fact that the transsys instance is copied is important, even though it is not relevant for understanding the `L-transsys` code shown in Fig. 2 (as the sole factor `A` is overwritten by the subsequent assignment in all `meristem` symbols).

Each rule in `L-transsys` is required to have a unique name (e.g. "`grow`" in the example discussed above). This allows keeping a record of all symbols generated during the growth of a simulated plant, including the `transsys` instance parameter of the symbol and the name of the rule which is activated. This allows to directly assess the correlation between gene expression levels, represented by `transsys` factor concentrations, and growth processes, represented by `L-transsys` rule executions.

[1]`http://www.cmp.uea.ac.uk/~jtk/transsys/`

```
transssys aux
{
  factor A
  {
    decay: 0.0;
    diffusibility: 0.0;
  }

  gene agene
  {
    promoter
    {
      constitutive: 1.0;
    }
    product
    {
      default: A;
    }
  }
}
```

```
lsys simplebrancher
{
  symbol meristem(aux);
  symbol shoot_piece;
  symbol left;
  symbol right;
  symbol [;
  symbol ];

  axiom meristem();

  rule grow
  {
    meristem(t) : t.A >= 5.0 -->
      [ left meristem(transssys t: A = 0.0) ]
      [ right meristem(transssys t: A = 0.0) ]
      shoot_piece meristem(transssys t: A = 0.0)
  }
}
```

Figure 2: The symbol, axiom and rule declarations of the L-transsys program. The symbol meristem is declared to be parameterised by the transsys program aux, shown left.

2.2 Objective Functions

The objective functions are designed to respond to correlations between gene expression levels and activation of growth rules. These are jointly captured in a growth record. Formally, the record of an L-transsys growth process is a set G of symbol records, where each symbol record is a tuple consisting of

- the time step
- the index of the symbol within the string
- the name of the symbol, denoted by s
- the name of the rule activated by the symbol, denoted by r
- the factor concentrations in the transsys instance parameterising the symbol, denoted by $c_1, \ldots, c_n$.

The subset of symbol records in which rule r is activated is denoted by G_r, and $\overline{G_r} = G - G_r$ denotes the subset of symbol records in which r is not activated. The set of concentration values of factor p within a set S of symbol instances is denoted by $C_p(S)$.

The set of expression levels found for factor p in symbols activating rule r is thus denoted by $C_p(G_r)$, and the expression levels of p in all other symbols are in $C_p(\overline{G_r})$. The suitability of p for determining activation of rule r can be quantified based on the disparity between these complementary sets: The greater the disparity, the better the factor is suitable for differentiating between symbols that should activate r and symbols that should not. The objective functions f_σ and f_{overlap} quantify this notion. Both objective functions are designed to evaluate to 0 if the gene product is differentially expressed such that it is can be used to accurately control activation of r. This case is referred to as perfect disparity. The objective function value is 1 if p is useless for determining activation of r.

For a factor p and a rule r, let $I_p^{r-} = [\min C_p(\overline{G_r}), \max C_p(\overline{G_r})]$ and $I_p^{r+} = [\min C_p(G_r), \max C_p(G_r)]$ and let $n_{\text{overlap}} = |\{c : c \in C_p(G), c \in I_p^{r+} \wedge c \in I_p^{r-}\}|$ be the number of expression level values of p which are in the overlap of the intervals I_p^{r+} and I_p^{r-}. The overlap objective function is defined as

$$f_{\text{overlap}}(r,p) = \tfrac{n_{\text{overlap}}}{|C_p(G)|} \tag{1}$$

The overlap objective function is 0 if there is no overlap of I_p^{r+} and I_p^{r-}. This means that either $\max C_p(\overline{G_r}) < \min C_p(G_r)$ (or $\max C_p(G_r) < \min C_p(\overline{G_r})$) and that, by choosing a threshold $t_{p,r}$ in the gap between the intervals, the condition $c_p < t_{p,r}$ can fully accurately control activation of r based on the expression level of p.

For the standard deviation objective function, let c_p^{r+} denote the mean expression level and σ_p^{r+} the standard deviation of c_p in symbols activating r, and c_p^{r-} and σ_p^{r-} denote the mean expression level and standard deviation of c_p in all other symbols, respectively. The standard deviation objective function is defined by

$$f_\sigma(r,p) = \frac{(\sigma_p^{r+} + \sigma_p^{r+})/|c_p^{r+} - c_p^{r-}|}{1 + (\sigma_p^{r+} + \sigma_p^{r+})/|c_p^{r+} - c_p^{r-}|} \tag{2}$$

The rationale here is that a small spread of values (small $\sigma_p^{r+} + \sigma_p^{r+}$) and a large difference of means ($|c_p^{r+} - c_p^{r-}|$) both are indicative of an accentuated disparity.

For both objective functions, the value for a rule r is defined as as the optimum among all factors:

$$f(r,P) = \min_{p \in P} f(r,p) \tag{3}$$

where P, the proteome, denotes a set of factors. The value for the entire growth record G is given by the mean value of the per-rule objective values:

$$f(G,P) = \frac{1}{|R|} \sum_{r \in R} f(r,P) \tag{4}$$

where R denotes the set of rules in the `L-transsys` system used to generate G.

2.3 Optimisation

The parameters that are subject to optimisation are the numeric values in a `transsys` program. Various `transsys` program elements contain an arithmetic expression,[2] see Fig. 1. Each arithmetic expression may be a tree of sub-expressions, where arithmetic expressions holding numeric values are terminals of this tree. The `transsys` framework has been extended to provide methods for extracting all arithmetic expressions that represent numeric values from a `transsys` program. The values stored in these arithmetic expressions are the parameters that are subject to optimisation.

A `transsys` program is evaluated by an objective function f by first implanting the program into an the `simplebrancher` system by merging its factors and genes into the `transsys` program

[2]Unfortunately, the term "expression" is used to refer to arithmetic expressions, i.e. formulaic prescriptions to compute numeric values, as well as to gene expression, i.e. the process of synthesising the product of a gene. To avoid ambiguity, the term "expression" is qualified as either arithmetic or pertaining to genes in this paper.

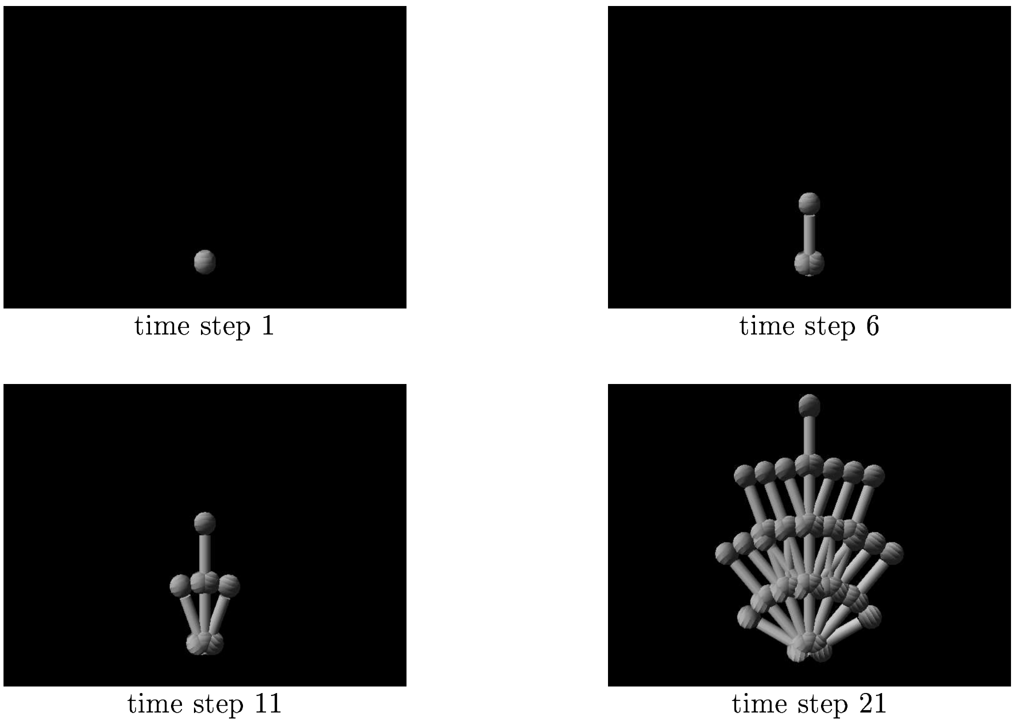


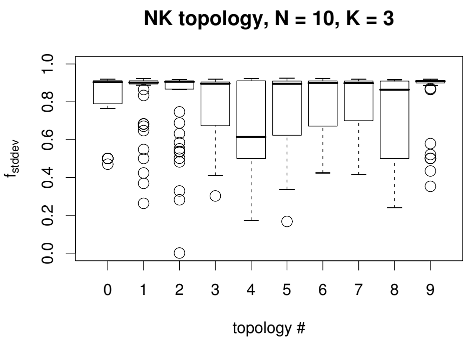
Figure 3: The developmental process used in the objective functions. The objective is to trigger the growth rule for branching every 5 time steps.

`aux`. The phenotypic growth process resulting from this system is identical to that observed with the unmodified `simplebrancher` code. After generating a growth record G of 21 time steps, the objective function value is given by $f(G, P)$, where P is the proteome of the implanted `transsys` program.

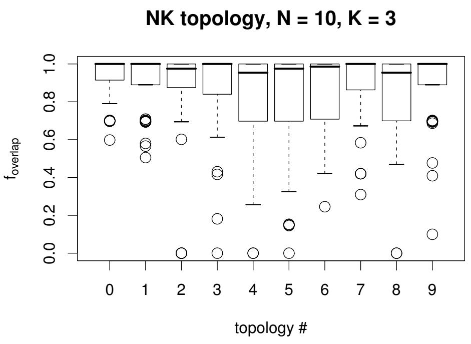
A simple, iterative local search approach is used to find a local minimum of the objective function. In each iteration, each parameter is offset by $\pm\delta$ while all other parameters are kept at their current values. Evaluations of the objective function in this δ–neighbourhood are used to estimate the local gradient. The current parameter vector is then updated by displacement along the estimated gradient. This procedure is iterated until improvement of the objective function falls below a threshold of 0.0001. For all results reported here, the offset is $\delta = 0.001$.

Initial tests showed that the random `transsys` programs could only be optimised to a limited extent using the overlap objective function. A major reason for this is that $I_p^{r+} \subset I_p^{r-}$ frequently occurs with random `transsys` programs because the rule r is activated only in relatively few symbols. If this is the case for all factors $p \in P$, $f_{\text{overlap}}(G, P) = 1$ and the gradient is flat, thus stalling the optimisation procedure. The problem local areas in which the overlap objective function is constant is further exacerbated by the fact that expression values have to move out of the overlap interval; just approaching its borders does not change the f_{overlap} result. This was one reason for choosing the relatively large offset of $\delta = 0.001$.

The standard deviation objective function is much less prone to stalling due to lack of local gradient information. Therefore, optimisation was performed sequentially by firstly optimising with f_σ and secondly subjecting the resulting `transsys` programs to optimisation with f_{overlap}. The rationale motivating this sequential optimisation procedure is that providing some degree of disparity using f_σ results in an increasing chance that the symmetric difference $I_p^{r+} \ominus I_p^{r-} \neq \emptyset$, thus providing a point of attack for optimising f_{overlap}.



NK topology, 30 edges, f_σ



NK topology, 30 edges, f_{overlap}

Figure 4: Optimisation results with the standard deviation objective function f_σ (left) and the overlap objective function f_{overlap} (right). Optimisation with f_σ was started from the value set inserted by the topology generator and from 50 random initial value sets. Optimisation of f_{overlap} was performed using the result of optimisation with f_σ as a starting point. Each box summarises the values of the objective after optimisation observed for 51 optimisation runs. Boxes depict the middle quartiles, whiskers extend to the most extreme value which is less than 1.5 times the interquartile range away from the box. Values outside the whiskers are depicted by individual points.

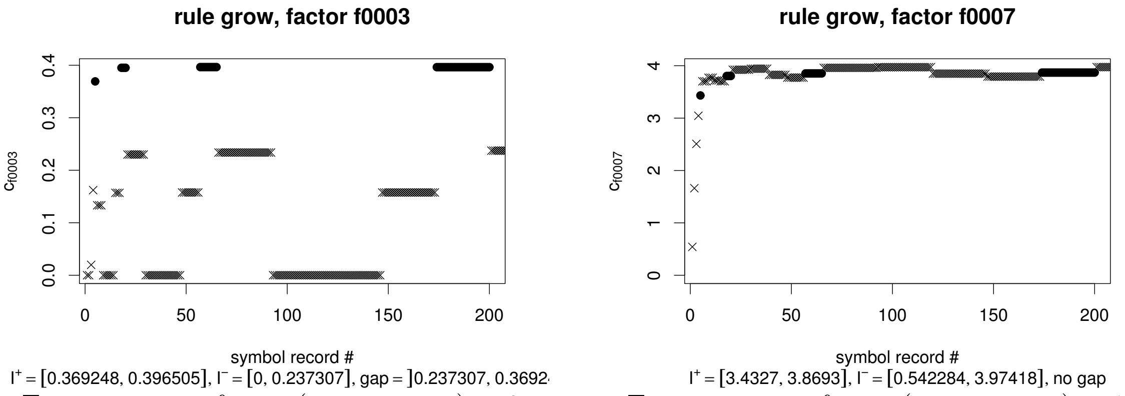
3 Results and Discussion

Random `transsys` programs were generated as described in [Kim, 2005]. NK graphs (in which all genes have the same in-degree K while the out-degree is subject to random variation, see [Kauffman and Weinberger, 1989]) and random graphs (Erdös–Rényi type in which both in- and out-degree are Poisson distributed) were used to generate the topology. The networks discussed here have 10 nodes (i.e. genes and gene products) were generated, with either 20, 30 or 40 edges, which, for the NK networks, corresponds to $K = 2$, $K = 3$ and $K = 4$, respectively. For each of the resulting 6 kinds of random `transsys` programs, 10 samples were generated. The numeric values in the `transsys` programs were initialised with random numbers drawn from a uniform distribution over$[0, 1[$, and the `transsys` programs were subsequently subjected to optimisation as described above. For each network topology, optimisation was performed starting from 51 different initial value sets.

Results of optimisation of transsys programs with NK topology and 30 edges are shown in Fig. 4. For all 10 topologies, the results with f_σ are subject to considerable variation, indicating a strong dependence of the outcome of optimisation on the starting parameter and thus presence of multiple local optima in the objective function. The results furthermore suggest that some `transsys` programs are more difficult to optimise than others, e.g. with topology #9, no substantial optimisation was achieved in the bulk of cases and there is only a small group of "outliers" for which $f_\sigma < 0.6$ is reached. In contrast to this, the median of f_σ is around 0.6 for topology #4.

The second stage of optimisation with f_{overlap} results in parameterisations for 5 out of the 10 topologies which produce perfect disparity. For the case of NK networks with 30 edges, a total of 10 such perfect networks are obtained (see Fig. 7).

Fig. 5 depicts a case of such perfect disparity. Factor `f0003` is expressed at levels in excess of 0.35 in all symbols that activate rule `grow`, while in all symbols that do not activate this rule, its expression level is below 0.24. For factor `f0007`, on the other hand, the range of expression levels in symbols activating `grow` is a subset of the range of expression levels found in symbols



Factor f0003: $f_{\text{overlap}}(\texttt{grow}, \texttt{f0003}) = 0$ Factor f0007: $f_{\text{overlap}}(\texttt{grow}, \texttt{f0007}) = 1$

Figure 5: Plots showing the disparity of gene expression of two factors in a $N = 10, K = 3$ transsys program obtained by sequential optimisation. Expression levels in symbols not activating rule grow are shown by crosses, levels in symbols activating grow are shown by filled circles. The order along the horizontal axis is not important (technically, it results from the temporal sequence of symbol string sequences), the relevant difference is that for f0003, threshold values between 0.24 and 0.35 are suitable to determine symbols activating grow.

not activating that rule.

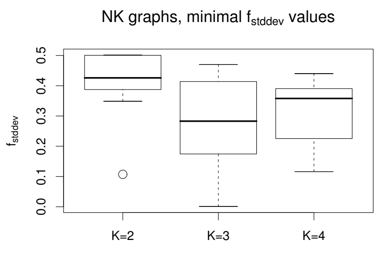
Surprisingly, analysing the complete data set obtained in this study do not show a significant effect of edge density on the performance of optimisation. However, differences are found when only the optimal results for each individual topology are considered. The results of this analysis are summarised in Fig. 6. For the NK topologies, increasing the number of edges from $K = 2$ to $K = 3$ results in some improvement of optimisation of both f_σ and f_{overlap}. For the random topologies, this trend is reversed, the best optimisation results are obtained with 20 edges and adding further edges reduces performance. The number of optimised networks that produce perfect disparity, shown in Fig. 7, appears to be correlated to the optimal values of f_σ.

The improvement seen with NK graphs with increased edge densities may be explained by the fact that more edges result in more optimisable parameters (i.e. a higher dimension of the search space), which may enable a kind of extra-dimensional bypassing [Cariani, 2002]. More specifically, the parameter optimisation procedure cannot insert new edges, but is capable of effectively removing edges by setting the maximum level of regulatory effect to 0. Therefore, the search space explored at higher edge densities implicitly contains subspaces that correspond to search spaces of networks with fewer edges. The decreased performance with high edge densities in random graphs may be due to the greater "disorder" in these graphs; while the in-degree in NK graphs is fixed to K, it is Poisson distributed in random graphs. Clearly, this is a tentative explanation and further analysis is required to understand the impact of random graph type on the structure of the objective functions and the resulting implications for the optimisation process.

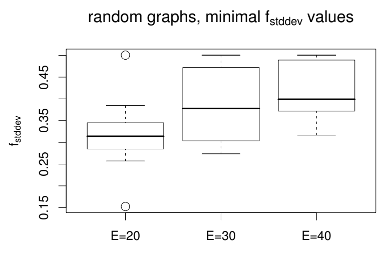
4 Conclusions and Outlook

This paper presents a method to parameterise RGNs based on the correlation of gene expression with the activation pattern of growth rules during a predetermined developmental process. As a proof of concept, this method was applied to obtain `transsys` programs that are suitable to drive the development of a simple branching structure.

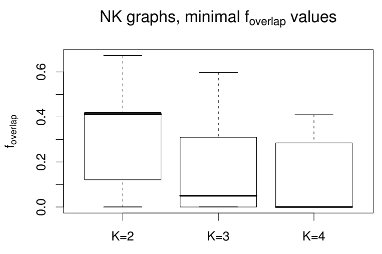
Developmental processes are organised based on genetic information, and consequently, computational models of development require an interface for transmitting information from the genetic



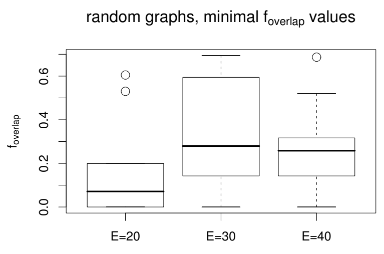
NK topology, f_σ



random topology, f_σ



NK topology, f_{overlap}



random topology, f_{overlap}

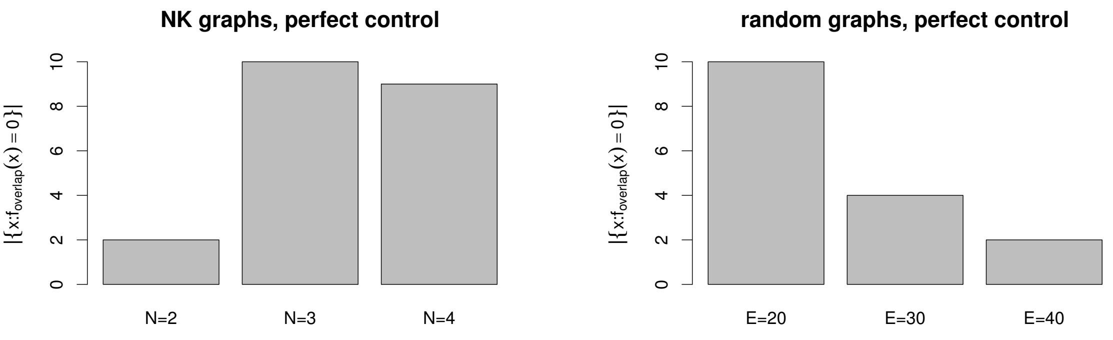
Figure 6: Boxplot of the minimal (optimal) values obtained with NK and random topologies and the f_σ and f_{overlap} objective functions.

level into the spatiotemporal process of development. Formal growth rules are a generic and fairly common element in Artificial Life models that link genetic information to development, and [Kniemeyer et al., 2004] demonstrates the potential of rule-based systems in Artificial Life modelling in general. The method presented here can therefore be applied with a large and general class of Artificial Life models. Synthetic RGNs obtained using this concept generally cannot be expected to model the corresponding molecular RGN in a sense of individual correspondences between synthetic and molecular RGN components. However, if network features exist that are required or favoured by the predetermined developmental process, they can be discovered by optimising correlations of gene expression and rule activation.

The work reported here is a step designed to enable new applications of the `transsys` framework for studying RGNs, their role in morphogenesis, and their evolution. As an immediate next step, the simulation of RGN reconstruction based on knockout mutants in [Kim, 2005] will be repeated with `transsys` programs obtained by the optimisation procedure introduced here, and controlling the `L-transsys` growth process, rather than using random `transsys` programs that receive information from the developmental process, but do not feed any information into its organisation. It will be interesting to explore whether comparing the original (wild type) symbol record with the records obtained with knockout mutants is useful to quantify the phenotypic effect of the knockout mutation.

The observed effects of edge density on optimisation performance with the two different types of network topology appear to be contradictory and are currently not satisfactorily explained. This highlights the currently limited understanding of the relationship between (static) network structure and regulatory dynamics. It might be interesting to use centrality measures [Koschützki and Schreiber, 2004] to further investigate this matter.

The objective functions introduced in this paper are multimodal and their optimisation is chal-



NK topologies — random topologies

Figure 7: Barplots of the number of optimisations yielding $f_{\text{overlap}}(G, P) = 0$, i.e. providing a factor that can fully accurately control the simulated plant's growth.

lenging, motivating the use of more advanced optimisation methods, such as Simulated Annealing. Following Artificial Life traditions, it is particularly attractive to invoke evolution for optimisation. To enable such approaches, a genome interpreter mechanism (see [Kim, 2000]), based on concepts similar to [Reil, 1999, Banzhaf, 2003, Schneider, 2000], has recently been added to the `transsys` framework. Mutating sequences that encode `transsys` programs may not only alter parameters but also introduce or delete genes and regulatory links. Therefore, this mechanism enables extension of optimisation to include the network structure in addition to its numeric parameters.

The results presented here are based on a very simple developmental process, and it can be expected that more complex developmental processes, involving more rules and comprised of more time steps and larger growth records, give rise to more demanding objective functions. Their optimisation can be tackled using an incremental approach in which the temporal length of the developmental process used in the objective function is gradually increased. This would allow the optimisation process to first achieve a good performance on the earlier stages of development without penalties for mistakes in the later stages, which might prevent the objective function from adequately reflecting success in early development. While this may render the optimisation procedure susceptible to committing itself to unfavourable local optima, exploring this approach using evolutionary methods for optimisation is attractive from a biological perspective. Biological development may indeed evolve in a comparable incremental way, which may be one of the roots of Haeckel's "biogenetic law" [Theißen and Saedler, 1995].

References

[Banzhaf, 2003] Banzhaf, W. (2003). On the dynamics of an artificial regulatory network. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J. T., and Ziegler, J., editors, *Advances in Artificial Life (ECAL 2003)*, volume 2801 of *Lecture Notes in Artificial Intelligence*, pages 217–227, Berlin Heidelberg. Springer Verlag.

[Barabási and Oltvai, 2004] Barabási, A.-L. and Oltvai, Z. N. (2004). Network biology: Understanding the cell's functional organization. *Nature Reviews Genetics*, 5:101–113.

[Basso et al., 2005] Basso, K., Margolin, Adam A. anad Stolovitzky, G., Klein, U., Dalla-Favera, R., and Califano, A. (2005). Reverse engineering of regulatory networks in human B cells. *Nature Genetics*, 37:382–390.

[Bentley and Clack, 2005] Bentley, K. and Clack, C. (2005). Morphological plasticity: Environmentally driven morphogenesis. In Capcarrere, M., Freitas, A. A., Bentley, P. J., Johnson, C. G., and Timmis, J., editors, *Advances in Artificial Life (ECAL 2005)*, volume 3630 of *Lecture Notes in Artificial Intelligence*, pages 118–127, Berlin Heidelberg. Springer Verlag.

[Cariani, 2002] Cariani, P. A. (2002). Extradimensional bypass. *BioSystems*, 64:47–53.

[Fleischer and Barr, 1993] Fleischer, K. and Barr, A. H. (1993). A simulation testbed for the study of multicellular development: The multiple mechanisms of morphogenesis. In Langton, C. G., editor, *Artificial Life III*, pages 389–416, Redwood City, CA. Addison-Wesley.

[Kauffman, 1987] Kauffman, S. A. (1987). Developmental logic and its evolution. *BioEssays*, 6:82–87.

[Kauffman and Weinberger, 1989] Kauffman, S. A. and Weinberger, E. W. (1989). The NK model of rugged fitness landscapes and its application to maturation of the immune response. *J. Theor. Biol.*, 141:211–245.

[Kim, 2000] Kim, J. T. (2000). Lindevol: Artificial models for natural plant evolution. *Künstliche Intelligenz*, 1/2000:26–32.

[Kim, 2001] Kim, J. T. (2001). `transsys`: A generic formalism for modelling regulatory networks in morphogenesis. In Kelemen, J. and Sosík, P., editors, *Advances in Artificial Life (ECAL 2001)*, volume 2159 of *Lecture Notes in Artificial Intelligence*, pages 242–251, Berlin Heidelberg. Springer Verlag.

[Kim, 2005] Kim, J. T. (2005). Effects of spatial growth on gene expression dynamics and on regulatory network reconstruction. In Capcarrere, M., Freitas, A. A., Bentley, P. J., Johnson, C. G., and Timmis, J., editors, *Advances in Artificial Life (ECAL 2005)*, volume 3630 of *Lecture Notes in Artificial Intelligence*, pages 825–834, Berlin Heidelberg. Springer Verlag.

[Kniemeyer et al., 2004] Kniemeyer, O., Buck-Sorlin, G. H., and Kurth, W. (2004). A graph grammar approach to artificial life. *Artificial Life*, 10(4):413–431.

[Koschützki and Schreiber, 2004] Koschützki, D. and Schreiber, F. (2004). Comparison of centralities for biological networks. In Giegerich, R. and Stoye, J., editors, *Proceedings of the German Conference on Bioinformatics (GCB 2004)*, pages 199–206, Berlin Heidelberg. Springer Verlag.

[Mendes et al., 2003] Mendes, P., Sha, W., and Ye, K. (2003). Artificial gene networks for objective comparison of analysis algorithms. *Bioinformatics*, 19:ii122–ii129.

[Prusinkiewicz and Lindenmayer, 1990] Prusinkiewicz, P. and Lindenmayer, A. (1990). *The Algorithmic Beauty of Plants.* Springer-Verlag, New York.

[Reil, 1999] Reil, T. (1999). Dynamics of gene expression in an artificial genome – implications for biological and artificial ontogeny. In Floreano, D., Nicoud, J.-D., and Mondada, F., editors, *Advances in Artificial Life*, Lecture Notes in Artificial Intelligence, pages 457–466, Berlin Heidelberg. Springer-Verlag.

[Repsilber and Kim, 2003] Repsilber, D. and Kim, J. T. (2003). Developing and testing methods for microarray data analysis using an artificial life framework. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J. T., and Ziegler, J., editors, *Advances in Artificial Life (ECAL 2003)*, volume 2801 of *Lecture Notes in Artificial Intelligence*, pages 686–695, Berlin Heidelberg. Springer Verlag.

[Rung et al., 2002] Rung, J., Schlitt, T., Brazma, A., Freivalds, K., and Vilo, J. (2002). Building and analysing genome-wide gene disruption networks. *Bioinformatics*, 18:S202–S210.

[Schneider, 2000] Schneider, T. D. (2000). Evolution of biological information. *Nucleic Acids Research*, 28:2794–2799.

[Theißen and Saedler, 1995] Theißen, G. and Saedler, H. (1995). MADS-box genes in plant ontogeny and phylogeny: Haeckel's 'biogenetic law' revisited. *Current Opinion in Genetics and Development*, 5:628–639.

[Van den Bulcke et al., 2006] Van den Bulcke, T., Van Leemput, K., Naudts, B., van Remortel, P., Ma, H., Veschoren, A., De Moor, B., and Marchal, K. (2006). SynTReN: A generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC Bioinformatics*, 7:43.

[Williams and Martinez, 2000] Williams, R. J. and Martinez, N. D. (2000). Simple rules yield comples food webs. *Nature*, 404:180–183.

[Wilson, 1989] Wilson, S. W. (1989). The genetic algorithm and simulated evolution. In Langton, C. G., editor, *Artificial Life I*, pages 156–166, Redwood City, CA. Addison-Wesley.

Evolutionary Robustness of Differentiation in Genetic Regulatory Networks

Johannes F. Knabe[1], Chrystopher L. Nehaniv[1,2], Maria J. Schilstra[2]

Adaptive Systems[1] and BioComputation[2] Research Groups
University of Hertfordshire
Hatfield AL10 9AB, UK
{j.f.knabe, c.l.nehaniv, m.1.schilstra}@herts.ac.uk

[A]ll cells of a given individual organism inherit the same set of blueprints in the form of DNA molecules. But as a higher organism develops from a fertilized egg a striking variety of different cell types emerges. Underlying the process of development is the selective use of genes, the phenomenon we call gene regulation. [... D]epending in part on environmental signals, cells choose to use one or another developmental pathway. - M. Ptashne [11, p. 1]

Abstract

We investigate the ability of artificial Genetic Regulatory Networks (GRNs) to evolve differentiation. The proposed GRN model supports non-linear interaction between regulating factors, thereby facilitating the realization of complex regulatory logics. As a proof of concept we evolve GRNs of this kind to follow different pathways, producing two kinds of periodic dynamics in response to minimal differences in external input. Furthermore we find that successive increases in environmental pressure for differentiation, allowing a lineage to adapt gradually, compared to an immediate requirement for a switch between behaviors, yields better results on average. Apart from better success there is also less variability in performance, the latter indicating an increase in evolutionary robustness.

1 Introduction

Typically in multicellular organisms, (almost) all of an individual's cells contain the same genome but still, depending on signals or differences in the internal environment, can take very different functional roles. Crucial signals are believed to be induced by other cells or the environment early in development, e.g. turning on (a) *homeotic* gene(s), which "remain on through adult life and maintain particular aspects of the pattern of gene expression characteristic of that segment [they are part of.]" [11].

In biological Genetic Regulatory Networks (GRNs), genes encode proteins and proteins in turn regulate the expression (activation) level of genes. The dynamics of these interactions not only play a key role in development [4] but also in the ongoing metabolism of all cells during their lifetime [1]. Furthermore, cells do not exist in isolation but are embodied in an environment, which influences the cell; the cell can in turn influence its environment via internal regulatory dynamics; see fig. 1.

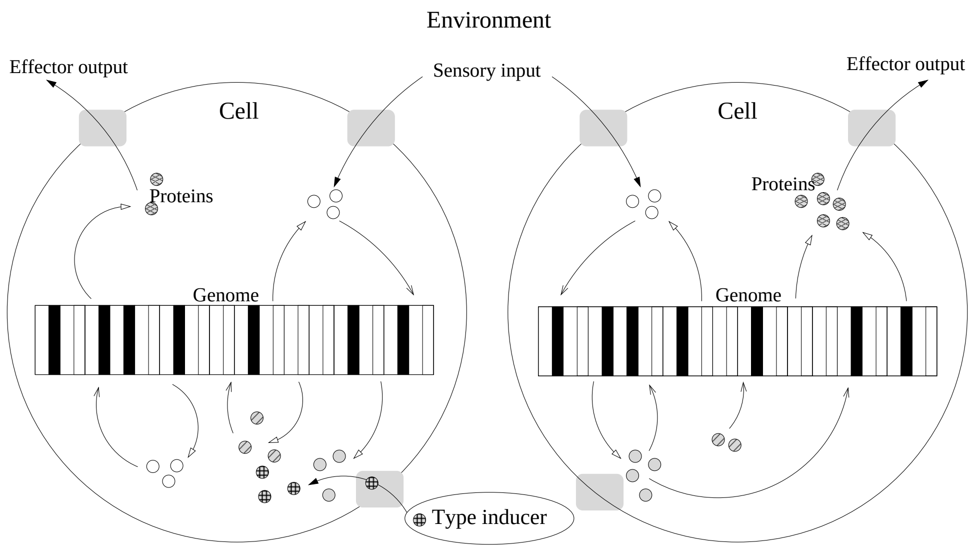


Figure 1: Schematic drawing of our model. The two cells have the same genome and thus the same regulatory network but can produce very different behavior, induced by a very simple signal which is here shown as external, but it could also be an internal gene that is always on due to cell division disparity.

As an evolutionary and computational paradigm, GRNs support complex regulatory and evolutionary dynamics [2], which when combined with differentiated multicellularity represent a vast potential for massive adaptive parallel and distributed computation [9]. This is achieved by a continual coupling of internal and external dynamics as active, regulatory control systems [12]. Differentiation of cells into types has been investigated in artificial GRNs several times. The most famous example is from Kauffman [7], but this and other early models are usually based on random boolean networks. Newer non-boolean approaches mostly have a strong pre-specification of the network structure (e.g. [5]), in this work we start from randomly connected networks.

2 Methodology

Before complicating matters by modeling huge multicellular structures with a diversity of cell types we begin with evolving a system capable of showing two behaviors. In [8], where we first described the proposed GRN model, we used it to evolve biological clocks with the circadian rhythm abstracted to a sinusoidal wave. GRNs producing such cyclic behavior in response to various periodic environmental stimuli could easily be evolved. Mirroring the phase of their input as well as the production of the inverse phase was possible[1], however with every evolutionary run having only one of these behaviors as its objective. So in the context of differentiation it was quite natural to ask whether it would be possible to integrate two or more functionalities into one GRN. We evolve populations of GRNs with two such functionalities in various settings and investigate the impact of the lineage's history on regulatory and evolutionary dynamics.
Cell cleavage and development are also victims of abstraction – from the start we have two identical cells receiving the same periodic external stimuli, cf. fig. 1. The expected difference in behavior is only signaled by a type inducer that raises a protein level, which in our model could be the result of either an internal gene turned on during cell division or externally generated. There is currently no diffusion or other kind of interaction between the cells.

[1]For results from those experiments see also http://homepages.feis.herts.ac.uk/∼kj6an/GRNclocks/.

2.1 GRN Model

The proposed GRN model makes locally smooth regulatory and evolutionary dynamics possible, and environmental interaction is explicitly considered. It has been first described in [8], where more details can be found.
Every cell consists of proteins and a genome with a fixed number of genes. Gene activation is controlled by regulatory sites (cis-sites or cis-modules), each composed of – possibly – several protein binding sites. Depending on the attachment of matching proteins to the binding sites the corresponding cis-modules positively or negatively influence the production of, not necessarily different, proteins. In molecular biology, proteins acting in such a way are called Transcription Factors (TFs). In our model all proteins are potentially regulatory. For simplicity in the regulatory dynamics we use template matching, i.e. a perfect match of binding site and the corresponding protein is required, unlike e.g. [2, 3]. The main difference to the `Biosys` model, described in [12], is that one can have any number of cis-modules per gene and every cis-module can have any number of protein binding sites. This is to allow for an additional level of protein regulation, as it is known to molecular biologists that TFs not only show additive behavior but might also interact with each other and thereby change their influence synergistically, see e.g. [13, and references therein]. This level could for example facilitate the advent of "master control genes", i.e. one active gene at the top of a control hierarchy that might start a cascade, turning on a huge number of other genes. For example [6] found that the out-of-place eye production in the fruit fly Drosophila can be triggered by a single signal. Such selectors can be thought of as choosing a particular pathway for the cell (as well as it's descendants) and are assumed to be involved in cell differentiation as well as developmental modularity.
In summary our approach facilitates the evolution of complex dynamics, coming a little closer to nature, where "5-10 regulatory sites are the rule that might even be occupied by complexes of proteins" [2].

2.1.1 Genetic Representation

The genome is represented as a string of integers, encoding the genes and some global parameters of the network. Digits 0 and 1 are *coding* digits that may be involved in regulation or protein coding. To differentiate between such a coding bit, a cis-module boundary and a gene boundary the genetic alphabet was increased to four digits, with 2 delimiting the end of a cis-module and 3 delimiting the end of a gene. There are eight different proteins in the version of the model used here, i.e. three bits encode a protein.
For this set of experiments we used a fixed number of genes, namely nine, as this had proven more than enough for coping with the single task described in our earlier paper [8]. After compartmentalizing the genome into genes, the last four coding digits of every gene determine its output behavior, three bits for the protein produced and the last bit for the gene's activation type, which can be "default on" – even active when no activation is present or "default off" – only with positive activation.

For cis-modules the first coding bit determines its influence on the gene's activation level (*inhibitory/activatory*) and every following three coding digits are considered a protein binding site. For example the gene 01011102110102001111 3 will produce protein 7 (111) and is "off by default" (last bit is 1). It has two cis-modules, the first inhibitory (starting with 0) binding a combination of proteins 5 (101) and 6 (110), and an activatory cis-module (starting with 1) to which protein 5 (101) will bind. Note that the last zero of 110102 is ignored; we refer to such coding digits which are neither translated nor regulatory as *junk*.
The genome also encodes several evolvable variables global to the cell. These are the *protein-specific decay rates* (four bit for every protein, indexing into a fixed lookup table of values), the global *binding proportion* (also four bits indexing into a lookup table, but identical for all

proteins), and finally the global *saturation value* (three bits indexing to a look up table, again identical for all proteins).

2.2 Regulatory Logic

The model is run over a series of discrete time steps, its lifetime. In each time step initially a fraction of the free proteins, determined by the global binding proportion parameter, are bound to matching sites; if there is more than one binding site competing for the same protein the fraction is equally distributed between all matching sites[2]. In this process all protein binding sites are treated equally, regardless of the cis-module to which they belong. Let b_i be the number of all binding sites matching protein i (there can be several for the same protein within and between cis-modules) and c_i^t denote the number of protein i being available for binding at time t. Then the amount p_{ijm}^t of protein i bound at time t to a given binding site in cis-module j of gene m and matching protein i is:

$$p_{ijm}^t = \frac{c_i^t}{b_i} + p_{ijm}^{t-1},$$

where p_{ijm}^{t-1} is the amount of protein i at the binding site in the previous timestep after saturation and protein-specific decay have been taken into account, with the initial condition $p_{ijm}^0 = 0$. The activation level a_m of gene m with k cis-modules is calculated as:

$$a_m = \sum_{j=1}^{k} \pm_j \min_{i:\ \text{protein } i \text{ binds to cis-module } j} p_{ijm}^t,$$

where $\pm_j = \begin{cases} +1 & \text{if cis-module } j \text{ is activatory} \\ -1 & \text{if cis-module } j \text{ is inhibitory.} \end{cases}$

Note that this use of min is similar to a logical AND and results in non-additive effects ("synergy") in gene regulation.

So the calculation of every gene's activation level is done by adding (activatory) or subtracting (inhibitory) the values per cis-module but only the lowest value of bound protein per cis-module is used (min). The increase in protein concentration due to gene m is then $f_m(a_m)$, [3] where

$$f_m(x) = \begin{cases} \frac{r}{2}\,(\tanh(\frac{x-15}{s}) + 1) & \text{if gene } m \text{ is "default off"} \\ \frac{r}{2}\,(\tanh(\frac{x+5}{s}) + 1) & \text{if gene } m \text{ is "default on".} \end{cases}$$

The parameter $s = 5$ determines the steepness of the slope, with the function becoming more switch like as s gets smaller, and $r = 150$ determines the range of the function. The output of the gene's activation function is added to the unbound concentration of that gene's output protein type. After this calculation the concentrations of all unbound proteins are, if necessary, reduced to the global saturation value and then all proteins, free or bound, are decayed by the protein specific rate. Finally environmental input occurs by increasing the unbound concentration of certain proteins by some value and output by reading some protein concentration values. Simple scaling by r is used to map stimulus input levels from the signal range to a protein concentration, and *vice versa* for output protein levels.

[2]Note that all variables for protein amounts are continuous.

[3]For example, for the gene 010111021101020011113 from above this would mean that due to the first (inhibitory) cis-module, assuming a share of 20 type 5 proteins (101) and 1 type 6 protein (110) per binding site, the value -1 would go into the sum. The second (activatory) cis-module however would contribute $+20$ resulting in an overall activation of 19, which gives a protein output of about 125 type 7 proteins.

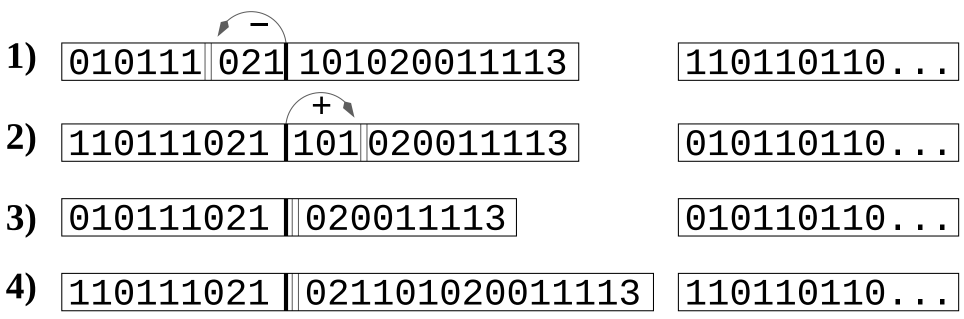


Figure 2: **Gaussian offset crossover.** Genomes of (1) parent 1, (2) parent 2, (3) offspring 1, (4) offspring 2. Only one gene and part of the global compartment shown. Both children get digits up to the crossover point from their respective parent, but then continue in the other parent's genome with opposite gaussian-distributed offsets (-3 and $+3$, respectively, here).

2.3 Evolution

We use a fairly standard Genetic Algorithm with weak elitism, tournament selection and replacement. Every evolutionary condition was studied with ten repetitions; each lasting 500 generations of 250 individuals, where one individual consisted of two cells with the same genome and thus the same regulatory network. The initial population started with one cis-module per gene and one protein binding site per cis-module, all coding bit values being randomly assigned; in network terms the nodes are randomly connected, with at most one incoming arc.

2.3.1 Selection

Later generations are formed by carrying over the best-performing individual of the last generation automatically and, keeping population size constant, the other individuals are replaced by offspring. To generate each pair of offspring, 15 (not necessarily different) individuals of the prior generation are chosen randomly and of these the best two selected to be "parents".

2.3.2 Variability

A (single-point) crossover between the parent genomes occurred 90 percent of the times and every coding bit is flipped with a mutation probability of one percent. To generate a variable number of cis- and of protein binding sites per gene it is necessary to have variable length genomes. Note that despite this, the number of genes stays the same all the time. These properties are achieved by dividing the parent genomes into compartments: one compartment for every gene and one compartment for the global variables. Then (with a probability of 0.9) a single compartment is chosen for crossover and in this compartment a point allocated for crossover. However when crossing over from parent 1's genome to the second parent's genome copying does not necessarily continue at the same position of parent 2's genome but is shifted by an offset (see fig. 2), mimicing the unequal crossing-over observed in biology.

This offset is randomly drawn from a gaussian distributed random variable with mean 0 and standard deviation 4. The relatively large number four was chosen to increase the chance of duplicating genetic information, the importance of which was already pointed out by [10] for the evolution of biological complexity. Ohno put emphasis on whole-genome duplications while it is now, with better techniques, becoming ever clearer that "both small- and large-scale duplication events have played major roles" [14].
Note that the offset point is limited to stay within the boundaries of the compartment, hence

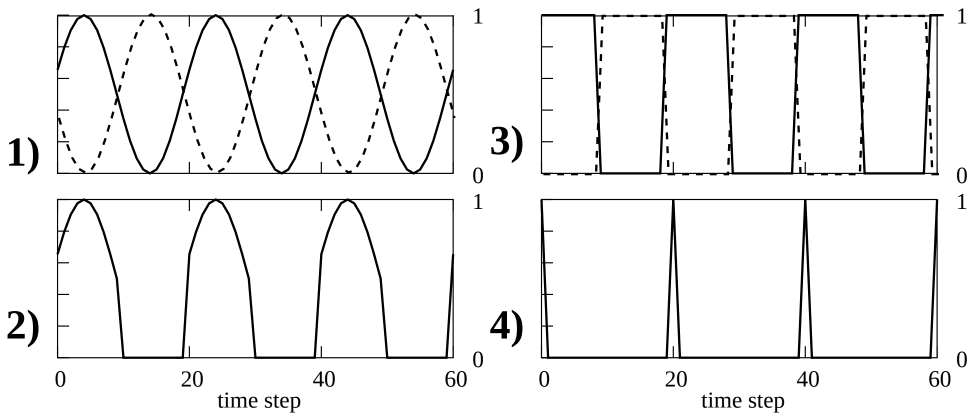


Figure 3: Periodic functions used: 1) sine (dashed the inverse/shifted wave), 2) positive part of sine, 3) step (dashed the inverse/shifted wave), 4) pulse.

if crossoverpoint + offset is smaller/larger than the left/right boundary it is set to the corresponding boundary value. So the number of 2s (cis-modules) might increase by crossover – mutation was only applied to coding digits (0s and 1s) – but not the number of 3s as these are the compartment boundaries. When crossover occurs in the part encoding for global parameters the offset is always set to 0 as more bits would be meaningless here.
These processes allow both neutral crossover and mutational changes, as 'half' cis-modules (i.e. less than three bit – one protein – long) are ignored. Additionally this means that, although the number of genes was constant over one evolutionary run, genes could become inactive, in a similar manner to the so called pseudo-genes found in nature, i.e. if there was not a single cis-module and the gene had an activation type of "off by default".

2.4 Environmental Coupling

We decided to systematically vary evolutionary conditions by varying the pattern of external signal received at the cellular level as well as the periodic output behavior expected.

2.4.1 Input stimuli

The basic idea was to have periodic environmental stimuli based on a sine curve (shifted to the interval $[0, 1]$). The wavelength w was set to 20 time steps, while the lifetime for every GRN was 400 steps. Variations included having only the positive part of sine, a periodic step function, and a brief pulse. The four functions used are depicted in fig. 3. As mentioned above, both cells of an individual always received the same periodic stimuli. However one cell additionally received an *inducing* signal with a continuous value of 1, realized as increasing the level of a protein type different from those used for periodic input and output.

2.4.2 Output behavior

Two periodic target functions were used to measure the performance of an individual and assign fitness: sine (fig. 3.1) and step (fig. 3.3). While the induced cell's desired output would be in the the same phase as the input, we ultimately want the other cell to produce the inverse of the input, which is equivalent to shifting the input's phase by one half. Fitness was measured as the deviation from this desired output, i.e. the smaller the value, the better adapted the GRN. Letting $c^t_{i_0}$ denote the (unbound) concentration of the induced GRN's output protein i_0 and d^t_p the desired output in phase p relative to that of the input at time t, the deviation is simply

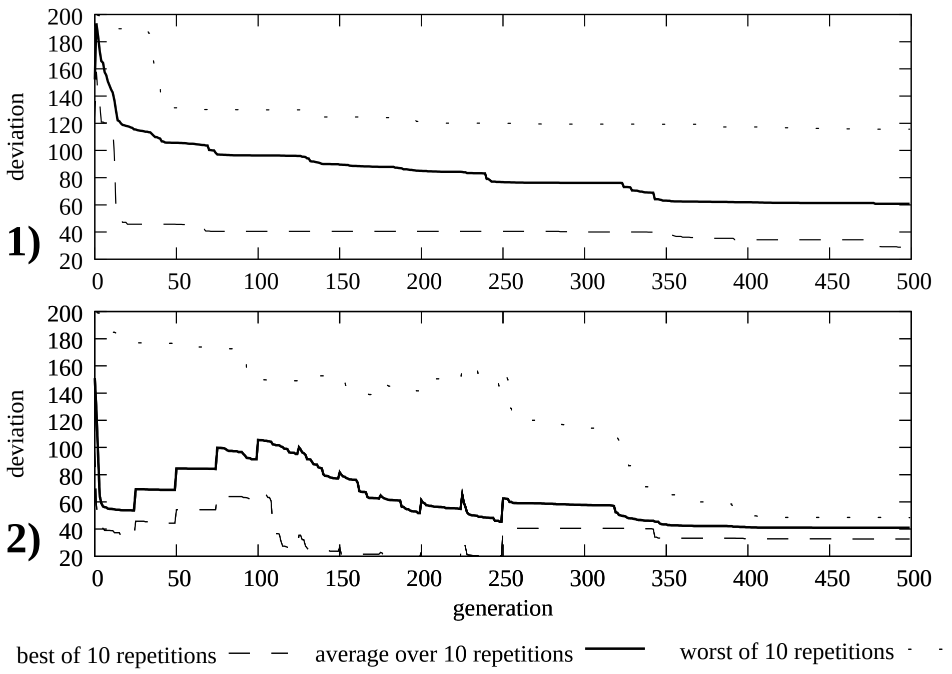


Figure 4: Exemplary evolutionary runs showing the best individual per generation (average over 10 repetitions); 1) with full differentiation pressure, 2) with gradually increasing differentiation pressure. For most experiments we found the best and worst repetitions to be closer together when the lineage's environment changed slowly.

calculated as: $\sum_{t=1}^{L} |c_{i_0}^t - d_{0.0}^t|$ and similarly for the other cell, only with $d_{0.5}^t$ – a phase shift of one half which is equivalent to the inverse wave. Finally both deviations were added up and divided by 2. The lifetime L of every individual was set to 400 time steps; as a reference, over such a lifespan a random GRN achieved a deviation of approximately 200.
However in one set of experiments we did not immediately, i.e. from the first generation, expect individuals to fully differentiate and rate performance accordingly. Instead, the environment became *gradually* harder by increasing the relative shift in wavelength little by little from 0 to $w/2$ every 25 generations (with g the current generation we wanted an output of $d_{\min(g \mod 25, w/2)/w}^t$ for the second cell in each pair).

3 Results

Overall, 8 evolutionary scenarios were tested (two desired output types times four environmental stimulus input functions) and each scenario was run ten times. Additionally, the whole set of 8 scenarios was repeated for gradually increasing environmental pressure, as described above.

3.1 Evolutionary Dynamics

In every scenario most repetitions successfully produced well adapted individuals that had evolved a kind of switch, allowing them to behave very differently when an inducing stimulus was present. Not very surprisingly, the more sparse the input was the harder it was to reduce the deviation from the desired output wave. For the immediate full shifting set of experiments, when considering a deviation of 80 acceptable[4], in 30 (out of overall 80) repetitions no GRN in the population could be considered to have achieved an acceptable performance level.

[4]By experience we found that GRNs with a performance worse than 80 often were much better at one task than the other, i.e. no real differentiation had taken place.

For the gradual setting however, this failure happened only twice, and the superiority of this condition can also be seen from table 1. It seems that an evolutionary environment gradually introducing a requirement for a switch between behaviors facilitates differentiation, and the smaller standard errors suggest an increase in evolutionary robustness. This is also reflected by the finding that for most experiments the best and worst repetitions are closer together when the lineage's environment changed slowly; for an example see fig. 4.[5]

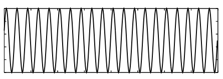
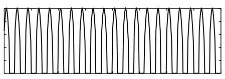
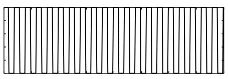
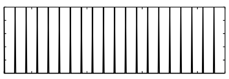
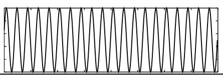
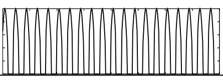
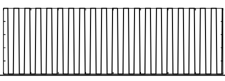
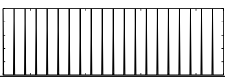
desired behavior / env. input	sine (inverse/mirror)	step (inverse/mirror)
sine 	**63.38** ±11.2 std. err. best evolved: 14.71 best rand.: 86.67/88.58	**76.19** ±12.0 std. err. best evolved: 27.90 best rand.: 126.2/92.01
pos. sine 	**50.14** ±6.29 std. err. best evolved: 21.78 best rand.: 80.18/75.37	**85.12** ±10.8 std. err. best evolved: 37.57 best rand.: 100.4/113.2
step 	**57.27** ±8.92 std. err. best evolved: 27.63 best rand.: 86.06/70.33	**60.75** ±9.40 std. err. best evolved: 28.90 best rand.: 72.17/70.84
pulse 	**74.34** ±6.25 std. err. best evolved: 27.93 best rand.: 86.44/89.68	**81.02** ±11.9 std. err. best evolved: 26.70 best rand.: 128.7/99.64
sine 	**29.52** ±3.62 std. err. best evolved: 18.87	**39.13** ±6.49 std. err. best evolved: 8.672
pos. sine 	**38.34** ±6.12 std. err. best evolved: 16.12	**56.34** ±6.83 std. err. best evolved: 31.04
step 	**37.15** ±3.10 std. err. best evolved: 24.78	**40.96** ±1.20 std. err. best evolved: 32.73
pulse 	**43.38** ±4.74 std. err. best evolved: 19.41	**63.59** ±6.66 std. err. best evolved: 23.39

Table 1: Outcomes of experiments with immediate (upper half), gradual (lower half) differentiation pressure, with the leftmost column depicting the environmental stimuli used and the topmost row the desired output behavior for every run. The data cells show the best final deviation averaged over 10 repetitions with 500 generations times 250 individuals each, ± the respective standard error. Additionally the best deviation achieved by evolution and, in the upper part, the best deviation found when testing the same number – 1.25 million – of random GRNs (one/two binding sites per gene are shown).

3.2 Evolved dynamics

In all the best evolved GRNs we found the use of AND-like regulatory logic with several binding sites bundled to a cis-module as described above, although the initial random nets started with only one site per module. Typically, the protein level being influenced by the type inducer, which might be considered as the output of a "master control gene" or an environmental stimulus, had a very prominent position (i.e. a high outdegree) in well adapted individuals. For example the one shown in fig. 5 participates in the regulation of 4 out of 7 functional genes. Finally, following

[5]Additional results as well as the full source code will be made available at http://panmental.de/GWALdiff

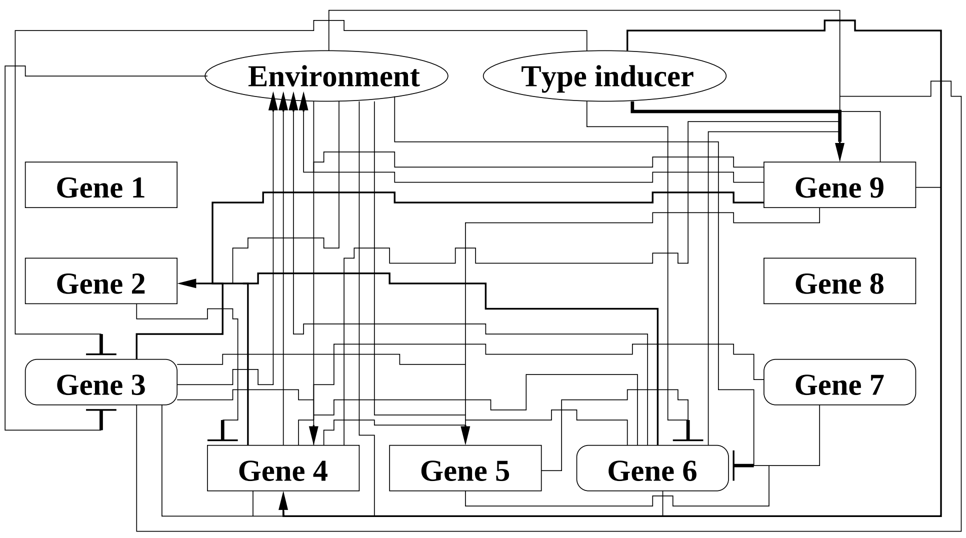


Figure 5: **Regulatory interaction diagram of an evolved 9-gene GRN.** Boxes denote genes (rounded corners indicating "default on" ones with the others being "default off"), connections ending in an arrow are for activatory influences and the T-like endings depict inhibitory ones. The bolder the connections the more binding sites the receiving gene has for the corresponding protein, resulting in a bigger share of the protein binding.

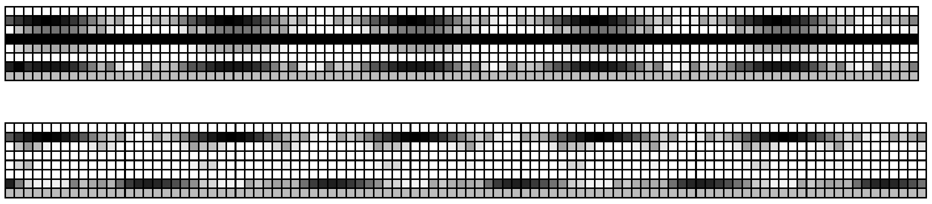


Figure 6: In these matrixes the **8 protein concentrations** of the GRN from fig. 5 over 100 time steps are depicted. Note that row 2 reflects the input protein level while row 7 corresponds to the GRN's output. In the lower matrix lack of activity in row 4 induces inversion of the input stimulus.

are figures illustrating what is going on in an exemplary individual which was the best of its repetition in the scenario: sine input, sine output desired, gradually increasing differentiation pressure. Figures 6 and 7 show its dynamics, with the lower matrices each corresponding to the "inverse desired" cell.

4 Discussion

The GRN model is clearly able to evolve functional differentiation. However the lineage's evolutionary history seems to be very important in determining the probability that a switch between two behaviors can be found. Comparing with the immediate requirement for a switch between behaviors we found that in the gradual case final GRNs usually showed better success with less variability in performance, the latter indicating an increase in evolutionary robustness. Similarities of input stimuli and expected output also had some impact, but differences were not huge and unsuprisingly the more sparse the input the worse the performance was.

In the future we will analyze the properties of evolved networks further – what do those that show a switching behavior have in common as opposed to those with no switch? – and also: How did the switch evolve? Last but not least, it will be interesting to see how well these findings

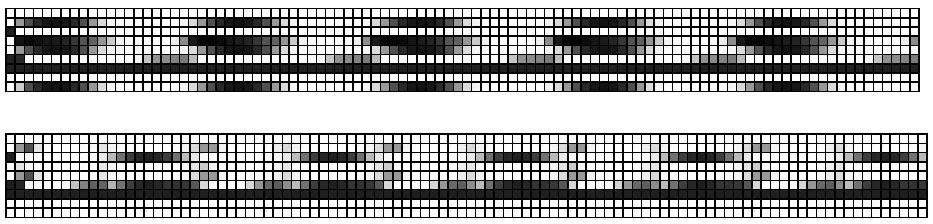


Figure 7: As in fig. 6, but here the **output activity of each of the 9 genes** is shown. Every row corresponds to one gene's protein output, where darker means more output. One can clearly see the distinct activation patterns. Note that genes 1 and 8 are inactive, i.e. generate no output ever, see fig. 5.

scale: can we evolve control hierarchies with levels of switching?

Acknowledgments

JFK wishes to thank Mark Robinson and Moritz Buck for useful discussions.

References

[1] Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, and Peter Walter. *Molecular Biology of the Cell*. Garland Science, 4th edition, 2002.

[2] Wolfgang Banzhaf. On the Dynamics of an Artificial Regulatory Network. In *Advances in Artificial Life, 7th European Conference, ECAL'03*, volume 2801 of *Lecture Notes in Artificial Intelligence*, pages 217–227. Springer, 2003.

[3] Peter J. Bentley. Adaptive fractal gene reguatory networks for robot control. In J. Miller, editor, *Workshop on Regeneration and Learning in Developmental Systems, Genetic and Evolutionary Computation Conference (GECCO 2004)*, 2004.

[4] Eric H. Davidson. *Genomic Regulatory Systems: Development and Evolution*. Academic Press, 2001.

[5] Nicholas Geard and Janet Wiles. A gene network model for developing cell lineages. *Artificial Life*, 11(3):249–268, 2005.

[6] G. Halder, P. Callaerts, and W. J. Gehring. Induction of ectopic eyes by targeted expression of the eyeless gene in drosophila. *Science*, 267(5205):1788–92, 1995.

[7] Stuart A. Kauffman. *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press, 1993.

[8] Johannes F. Knabe, Chrystopher L. Nehaniv, Maria J. Schilstra, and Tom Quick. Evolving biological clocks using genetic regulatory networks. In *Proceedings of the Artificial Life 10 Conference (Alife X)*. MIT Press, 2006 (in press).

[9] Chrystopher L. Nehaniv. Self-replication, evolvability and asynchronicity in stochastic worlds. In *Stochastic Algorithms: Foundations and Applications*, volume 3777 of *Lecture Notes in Computer Science*, pages 126–169. Springer, 2005.

[10] Susumu Ohno. *Evolution by Gene Duplication*. Springer, 1970.

[11] Mark Ptashne. *A Genetic Switch*. Cell Press and Blackwell Science, 2nd edition, 1992.

[12] Tom Quick, Chrystopher L. Nehaniv, Kerstin Dautenhahn, and Graham Roberts. Evolving Embodied Genetic Regulatory Network-Driven Control Systems. In *Advances in Artificial Life, 7th European Conference, ECAL'03*, volume 2801 of *Lecture Notes in Artificial Intelligence*, pages 266–277. Springer, 2003.

[13] Maria J. Schilstra and Hamid Bolouri. Modelling the Regulation of Gene Expression in Genetic Regulatory Networks. Technical report, BioComputation group, University of Hertfordshire. http://strc.herts.ac.uk/bio/maria/NetBuilder/Theory/NetBuilderModelling.htm, 2002.

[14] John S. Taylor and Jeroen Raes. *The Evolution of the Genome*, chapter Small-Scale Gene Duplications. Elsevier Academic Press, 2005.

Multi-Agent Design to reduce complexity of biological processes description

N. Parisey[1] M. Beurton-Aimar[1&2], C. Lales[2] and J.P. Mazat[1]

[1]Laboratoire de Physiopathologie Mitochondriale - INSERM U688 - University Bordeaux 2.
[2]Laboratoire Bordelais de Recherche en Informatique - CNRS UMR 5800 - University Bordeaux 1.

Abstract

Metabolic processes involve numerous enzymatic reaction pathways and constitute complex interaction networks. Most often each enzymatic reaction implies at least two metabolites (molecules) and the level of interactions depends upon each molecule current state. Even if participants exhibit a small number of different possible states, this can lead to a combinatorial explosion of the number of reachable states of the whole system. Using a multi-agent design it is possible to easily take into account this complexity. In such design, agents are enzymes or metabolites and each of them stores its own state. The complexity is then recovered during the computer simulation of the process. We have studied a specific case of metabolic process, called the Q cycle, which involves one enzyme and two types of metabolites. The Q cycle is the central part of the respiratory chain in mitochondria which mainly produces cell energy. Each metabolite can have two or three possible redox states depending on its type. The enzyme has six reactive sites and each site has two possible states : free or occupied by a metabolite. Taking into account all the possible states of the system leads to model several hundred system states. A traditional way in biology is to code the system by a set of differential equations. Nevertheless, we show that a multi-agent system can model this kind of process more easily than the traditional one. The obtained tool allows to manage and to modify conditions of simulations for testing different hypotheses on normal or pathological situations in an easy way.

Introduction

The last twenty years have seen the publication of many mathematical models to simulate biological processes. Usually, these models use Ordinary Differential Equations (ODE). ODE allows us to take into account the time course of molecule concentrations but large metabolic pathways rapidly become difficult to study. Moreover, ODE models lack the ability to efficiently deal with the localisation of molecules and the network topology parameters (compartments for instance). Partial Differential Equations (PDE) can be used with such problems but they are more difficult to handle. Finally, in differential equation models, some important data such as experimental conditions are often discussed in the text but do not appear explicitly in the mathematical model.

As part of a project to simulate the whole mitochondrial metabolism, we have studied a specific metabolic process involved in mitochondrial energy production. This process called Q cycle is a set of redox events that take place within the Respiratory Chain of the mitochondria. There already exists several ODE models of the Q Cycle [4] but they do not take into account the whole cycle behaviours because of the complexity level of this process. In order to propose a more complete model of the Q cycle, we have built a model using Multi-Agent design to implement molecule's behaviours.

Multi-Agents Systems (MAS) are currently used to investigate structures, behaviours and functions of metabolic [12] and biochemical [7] processes. Metabolic processes involve interactions of different entities, molecules and enzymes, within a situated physical environment. Biochemical reactions are discrete and local events that will create global behaviour and organisation within the metabolic system. The study of metabolic pathways involves the design of biochemical reaction chains between molecules. In silico, simulations of such reaction chains can help understanding normal or pathological cell behaviours.

The following sections present, firstly, the Q cycle process, from a biological point of view and from ODE point of view, and secondly the Multi-Agent model of this process. Finally, we discuss future works.

1 Mitochondrial Q cycle: energy synthesis

Mitochondria are intra-cellular organelle in eukaryotes. Each cell is filled with a set of hundreds of mitochondria. The essential role of mitochondria is energy generation for the cell. The energy delivered by mitochondria is the result of a complex mechanism, involving a series of redox reactions called *"Respiratory Chain"* which uses the oxygen we breathe in and the nutriments we ingest. A redox reaction is characterized by an electron exchange between two partners according to qualitative and quantitative rules. The Respiratory Chain redox reactions are catalysed by highly structured protein complexes and electron transporters embedded in the inner mitochondrial membrane. Electrons transfer is linked to an extrusion of protons (H+) outside the mitochondria, creating an electro-chemical gradient, which will be used, among others, for ATP synthesis, according to chemiosmotic theory [11]. As shown in Fig.1, this process involves the Respiratory Chain complexes (complex I to IV), an Iron Sulfur Protein (ISP) and ATP synthase (Complex V).

Complex III plays a central role in Respiratory Chain [11]. Like all the Respiratory Chain complexes, Complex III is embedded in the inner mitochondrial membrane. As shown in Fig.2 this complex catalyses the transformation of CoEnzyme Q from its more reduced form *ubiquinol* (UQH2 or CoQH2) into its more oxidized form *ubiquinone* (UQ or CoQ). The process is in fact fairly complex. Mitchell was the first to design the internal work of the complex III (concept in 1967, fully developed by 1975[9]). All the features that occur within the complex III were called the *Q cycle*. At present it is known that during the catalytic process, UQH2 turns into *semiubiquinone* ($UQ^{\bullet-}$ where $\bullet-$ represents an electron) before reaching UQ state. The CoEnzyme Q molecules in all their different redox states are collectively named the ubiquinone pool. So the UQH2, $UQ^{\bullet-}$ and UQ designate three different possible redox state of the Co Enzyme Q molecule.

The $UQ^{\bullet-}$ transition and other hypotheses have been included in the initial theory and the Q cycle has become the *modified Q cycle* [3] [1]. Due to the occurrence of the transient free radicals species ($UQ^{\bullet-}$), the respiratory chain is also the site of Reactive Oxygen Species (ROS) production [3]; this is a deleterious process, which can be amplified in mitochondrial diseases and aging.

The complex III (Fig.2) sustains the Q cycle mechanism. It is built of several subunits, among

[1]From now on, in the text, any reference to the Q cycle refers in fact to the modified Q cycle

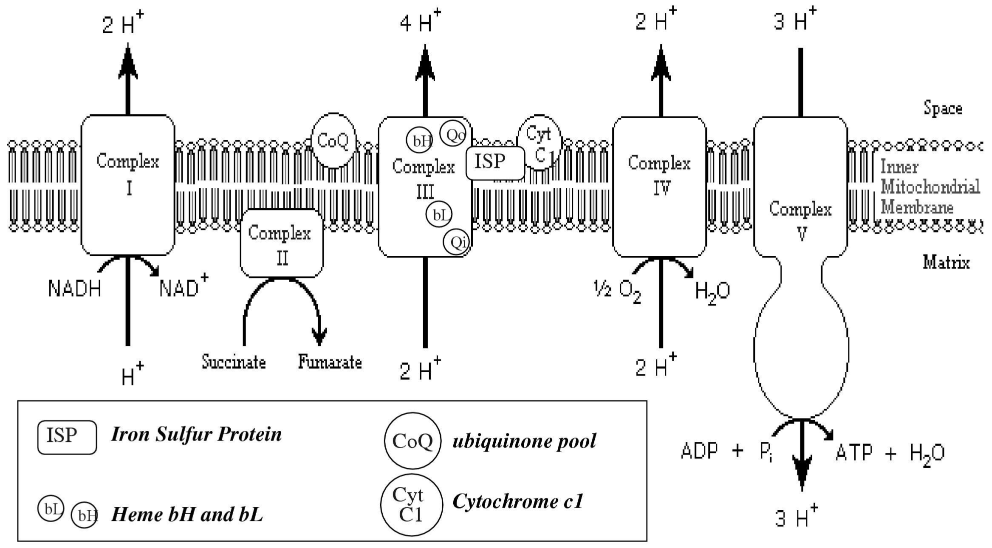


Figure 1: A simple representation of the Respiratory Chain and ATP synthesis

which the cytochrome b, the Iron Sulfur Protein and the Cytochrome c1. The complex III contains two binding sites (Qo and Qi) and two interacting sites: one is an heme of *Low redox potential* (bL) and the other is an heme of *High redox potential* (bH). Both hemes, which are part of the cytochrome b, have redox properties due to the presence of an iron molecule (Fe^{2+}/Fe^{3+}). The Qo site is near the outer side of the internal mitochondrial membrane, and the Qi site is facing the mitochondrial matrix. Qo and Qi sites can both bind UQ, $UQ^{\bullet-}$ or UQH2. Two other proteins will also play important roles in the Q cycle. The Iron Sulfur Protein has a site that contains an Fe-S centre. The Cytochrome c1 possesses a site that contains an iron molecule.

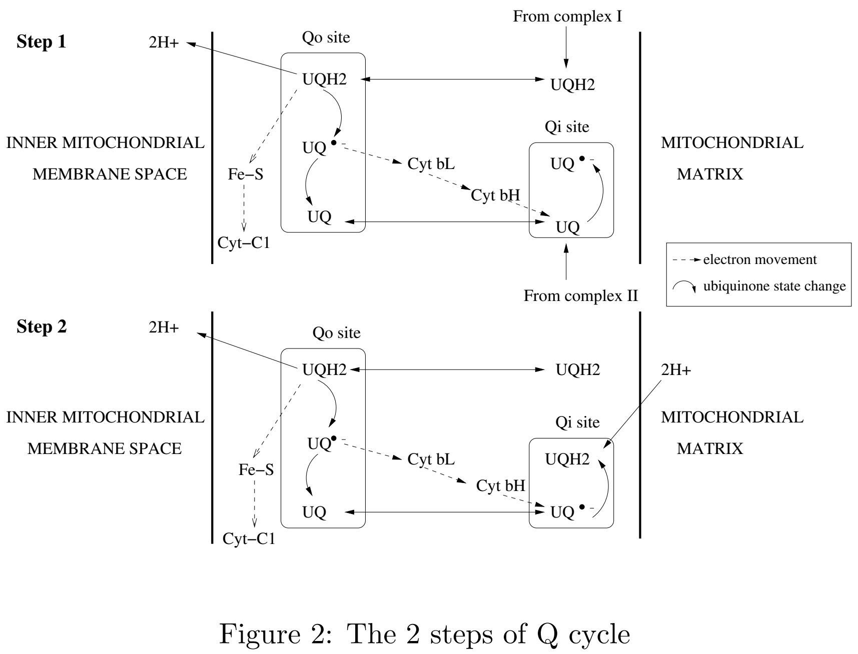


Figure 2: The 2 steps of Q cycle

A brief description of the Q cycle is given in figure 2. It consists of two steps.

Step 1: it starts with an electron being transfered from UQH2 (bound to the Qo site) to the *Fe-S* centre of an Iron Sulfur Protein (ISP). This triggers the release of two protons into the inter mitochondrial membrane space and transforms UQH2 into $UQ^{\bullet -}$ (still bound at site Qo). Next, an electron is transfered from the Fe-S to a Cytochrome c1 and the $UQ^{\bullet -}$ (becoming UQ) gives an electron to the Cytochrome b Low potential heme (*Cyt bL*). Electron transfer from Low potential heme to High potential heme (*Cyt bH*) directly follows. This electron is then given to an UQ molecule at Qi site thus forming $UQ^{\bullet -}$.

Step 2: it starts with the release of UQ from the Qo site. A new UQH2 molecule takes its place. The previously generated $UQ^{\bullet -}$ stays in the Qi site. After that, the scenario is identical to the first step but instead of a UQ, it is a $UQ^{\bullet -}$ that receives the electron from the Cyt bH, thus forming UQH2 (on Qi site) with two protons taken from the mitochondrial matrix.
To reach step 2 from step 1 implies that an UQ molecule binds Qi site and that a new UQH2 molecule binds to the Qo site.

1.1 Modeling the Q cycle with ODE systems

Several models have been proposed to simulate and investigate the properties of the Q cycle. Most of them are based on Ordinary Differential Equations (ODE). These models describe the time evolution of different redox forms of Coenzyme Q and of the sites states.

A case study ODE model [4] involves 31 variables, the variation of which are described by a set of 31 ODE combining 43 rate equations. Among them only 25 rate equations out of 43 code for the Q cycle itself. The other ones concern initial conditions, steady state constraints and output calculations.

Variables	Different States		
Fe	Fe-S	Fe-$S^{\bullet}$	Fe-$S^{\bullet}Q^{\bullet}$
Cyt bh	Cyt bH	Cyt $bH^{\bullet}$	
Cyt bl	Cyt bL	Cyt $bL^{\bullet}$	

The ODE model describes a composition of different states of 3 variables (table based on [4]).

Some key players of the Q cycle are left aside among this states: cytochrome c1 and ubiquinone at the Qi site. Also Fe-$S^{\bullet}Q^{\bullet}$ state is in fact Fe-$S^{\bullet}$ with $Q^{\bullet}$ fixed within the Qo site.

These three variables in their different status summarise different status of Qo and Qi sites.

Thus, there are 12 possible states (Fig.3) for the whole cycle. A total of 22 possible reactions describe the transitions from one state to another. They are tagged according to the step cycle in which they take place.
The 25 rate reactions are in fact derived from 7 type of generic reactions applied to different variables representing different redox states of the complex III. Thus it exists a certain amount of redundancy within this ODE model.

In a sense, numerical variables of the equations increase because of the spatial conformation and the number of the subunit states involved. This is a built-in feature of several ODE models. If the behaviour of each key players of the Q cycle were to be described one by one, the combination

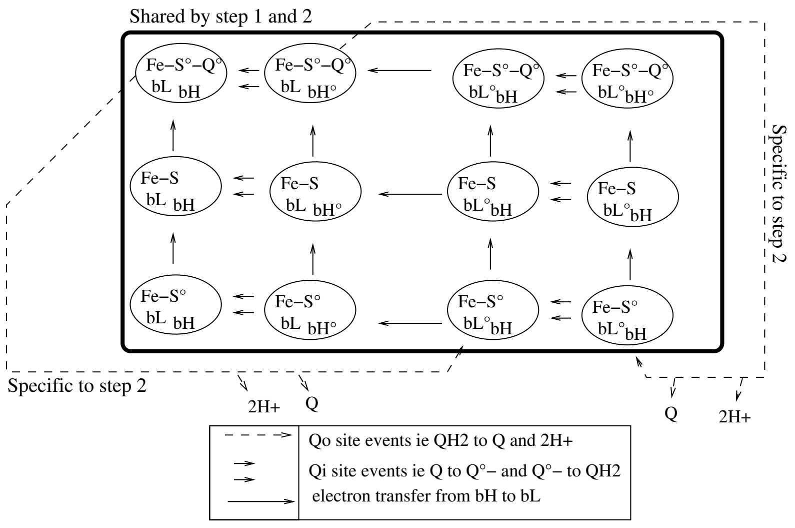


Figure 3: Graph of the 12 states and the 24 valid transitions (based on [4]

of their behaviours will automatically recreate all the states and transitions of the model, and probably some others relevant for pathologies.

2 Multi-agent models to simulate redox reactions

The dynamic of the system can be better designed using paradigms such as Multi-Agent Systems (MAS). There exists several examples of such MAS usage for living systems like artificial chemistry[2]. For a couple of years they have also been used in biology to simulate membrane generation [6] or biological pathways [2].

In MAS design, agents are entities which can interact with each other following rules. In our model we have chosen to set the granularity to the molecular level. One of the important differences between MAS model and ODE is that the former works with discrete values, number of molecules, whereas the latter uses continuous variables, such as species concentration (mole per litre). The calculation of ODE is always a calculation of averaged variables of the system, whereas that of MAS allows individualization of each molecule with its own behaviour. A MAS based on *reactive agents* is a simple way to represent metabolic processes where molecules are agents.

We will present the MAS model that we have designed for the Q cycle simulation.

2.1 Reactive agents and the Q cycle

An agents is characterised by its name (ie species name) and its position in a 3D space. The MAS has mainly two major types of agents: `Metabolite` and `Enzyme`. We have used a generic agent `BioAgent` that can handle common attributes (name, weight, position, structure). This class is subclassed into `Enzyme` and `Metabolite`. `Enzyme` and `Metabolite` are also subclassed to code specific type of enzyme or metabolite. The `Metabolite`s involved are `UQ`, `Fe-S` and `Fe`. The `complex III` is a subclasse of `Enzyme` agent.

[2]see www.alife.org for a complete list of links

In the MAS, a random walk in a 3D space is applied to each allowed agent and the affinity values between enzyme and metabolites are represented as catch/release probabilities.

An `Enzyme` is a `BioAgent` which main attribute is a list of sites. Those sites are affected affinities for `Metabolite`s (catch and release probabilities) and spatial positions. Each site has two states: free or occupied by a `Metabolite`. Sites are used by the `Enzyme` to modify the redox value of the `Metabolite`s clustered in them. In our approach, these sites are only attributes of the `Enzyme` agent since they are not independent and are modelled as elements of the `Enzyme`. For our application, complex III `Enzyme` has the following list of site: `bL`, `bH`, `ISP`, `Cytc1`, `Qo` and `Qi`. But if it is necessary an enzyme can become a composite agent containing some other `BioAgents`.

In the Q cycle model, we take into account two kind of sites: interacting and binding for the complex III. Interacting sites are `bH`, `bL`, `c1` and `Fe-S`, while binding sites are `Qo` and `Qi`. An **Interacting site** is an `Enzyme` site where a metabolite is definitively bound. This metabolite will follow redox rules. A **Binding site** is an `Enzyme` site where a metabolite can dock to the site and afterward be released.

Probabilitistic rules apply for describing metabolite binding and release on sites. In the case of `Fe` and `Fe-S`, the binding parameters lead to an irreversible link to their sites. A metabolite is considered bound when it is spatially close to the site. So the spatial position of metabolite can vary within a given range.

In classical biological descriptions, when an interacting site and its metabolite are definitely bound, they are considered merged together into one entity (the molecule that interacts is merged with the spatial site in which it is clustered). In the previous ODE model, bL and bH variables described the site within complex III and also the molecule clustered in it.

The Q cycle process is mainly driven by redox reactions. Thus in the MAS model, the different Metabolite interactions are described by redox rules. The next paragraph describes how we have modelled these redox rules.

2.2 Modeling redox rules

Redox rules represent the firing of redox reactions. Each `Metabolite` possesses a current state of oxidation (from a range of possible states). Changing its state is done using a threshold (a redox value) given by its `Metabolite` type and its current redox state. A redox reaction consists in switching its state for each participant (from oxidized to reduced or the opposite). When two `Metabolite`s are close, their redox values are compared: if the molecule with the greater redox value is in an oxidized state and the other one is in a reduced state then the redox reaction occurs.

For example, the reaction between UQH2 and Fe-S is:

- The redox value for the CoEnzyme Q to switch from its most reduced state (/`UQH2`) to a less reduced state ($\mathtt{UQ}^{\bullet-}$) is 170 mV while the redox value for the Iron molecule to switch from its an oxidized state ($\mathtt{Fe\text{-}S}_{ox}$) to a reduced state ($\mathtt{Fe\text{-}S}_{red}$) is 280 mV.
- As `UQH2` is the most reduced state of ubiquinone, it reacts with `Fe-S`, if `Fe-S` is in oxidized state, to give $\mathtt{UQ}^{\bullet-}$ on the one hand and $\mathtt{Fe\text{-}S}_{red}$ on the other hand. Thus the reaction

takes place in the direction: UQH2 + $\mathtt{Fe\text{-}S}_{ox} \rightarrow \mathtt{UQ}^{\bullet -}$ + $\mathtt{Fe\text{-}S}_{red}$.

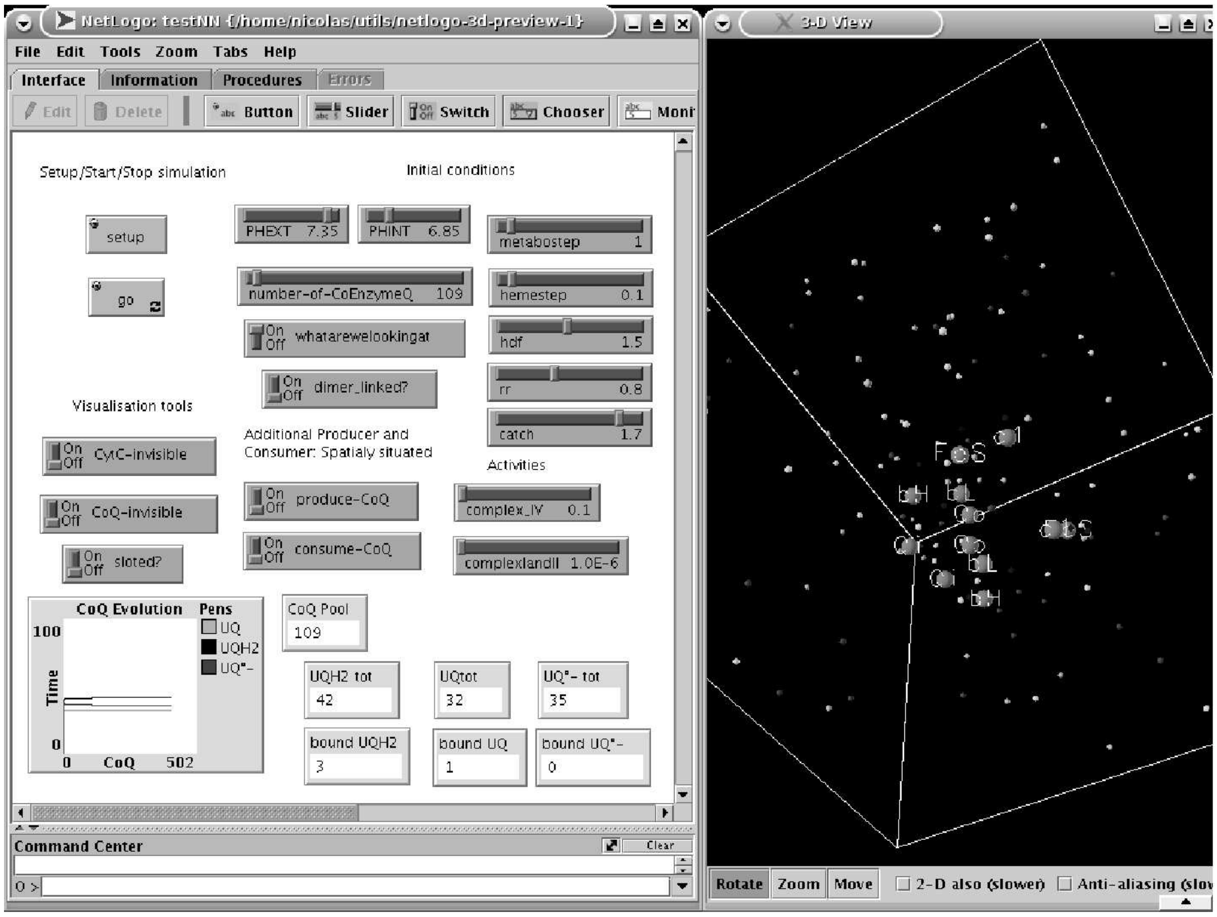
All the steps of the process done in Fig.2 obey the same rule.

BioAgent Life Cycle In a Multi-Agent design, the behaviour of all agents are driven by a Life Cycle. We have chosen to describe an asynchronised algorithm for this Life Cycle.

Each step of the algorithm is as follow:

- Each `BioAgent` in a randomised list is inspected,
- its new state is calculated, taking into account, its environment: the position (coordonates in a three dimensional continuous space) and the state of its neighbours. The `BioAgent` switchs from its current state to the new state.

A first prototype has been set up using a "cross-platform multi-agent programmable modelling environment", NetLogo [10].


NetLogo: testNN {/home/nicolas/utils/netlogo-3d-preview-1}
File Edit Tools Zoom Tabs Help
Interface Information Procedures Errors
Edit Delete Button Slider Switch Chooser Monit
Setup/Start/Stop simulation
Initial conditions
setup
go
Visualisation tools
Additional Producer and Consumer: Spatialy situated
Activities
CoQ Evolution
Pens
UQ
UQH2
UQ•−
Time
CoQ
Command Center
Clear
3-D View
FeS
bH
bL
Rotate Zoom Move
2-D also (slower) Anti-aliasing (slow

PHEXT 7.35
PHINT 6.85
metabostep 1
number-of-CoEnzymeQ 109
hemestep 0.1
On Off whatarewelookingat
hdf 1.5
On Off dimer_linked?
rr 0.8
catch 1.7
On Off CytC-invisible
On Off CoQ-invisible
On Off sloted?
On Off produce-CoQ
On Off consume-CoQ
complex_IV 0.1
complexIandII 1.0E-6
CoQ Pool 109
UQH2 tot 42
UQtot 32
UQ•− tot 35
bound UQH2 3
bound UQ 1
bound UQ•− 0

Figure 4: A prototype

This prototype (Fig.4) implements the Multi-Agent model. It simulates the behaviour of a `complex III` dimer [3]. It enables to follow the time course of metabolites concentration (by type

[3]that is: two `complex III` spatially linked to one another, not a composite `BioAgent`

and redox states). We have implemented procedures to track redox reactions which occurs for each `Metabolite`. This will permit us to analyse the redox reaction chains that follows the Q Cycle behaviours and those that are potential shortcuts.

Such MAS implementation allows to treat all the cases specified with the corresponding ODE model. Moreover, we are now able to simulate new combinations of parameters which can correspond to normal or pathological situations. For example, abnormal short circuit situations are allowed and can occur in our model. For instance, the electron fate after binding of UQH2 to the Qo site is not imposed.
Lastly, inclusion of this part of the respiratory chain into a more complete model of the mitochondrial metabolism is not a problem. The composition of several models is obtained by writing interface (rules of exchange) between the elements of the composition. Connection of `complex III` with `Complex I` or `II` on the one hand and with the `Complex IV` on the other hand is given by the rules of production or consumption of `UQH2` and `UQ` molecules respectively.

3 Discussion and perspectives

The description of Q cycle has evolved over time since the first propositions by Mitchell [9], particularly after obtaining the 3D crystal structure of complex III [5]. Descriptions of molecule behaviours and sequence processes are experimentally more and more detailed. As a consequence the number of ODEs increases. The ODE users thus have the tendency to minimise the number of process taking into account, by ignoring the apparently less frequent (less probable) reactions.

On the contrary, the MAS model leaves the system free to simulate all events just by following the rules of interactions between agents. MAS can be considered the right level of abstraction for designing and engineering complex systems characterised by organisational structures and coordination processes [13]. In our example, representing the 12 states and the 22 transitions given in Fig.3 is possible with the definition of two generic types of entities (`Enzyme` and `Metabolite`), a list of ordered states and a list of redox values for each given enzyme and metabolite, and a generic rule describing redox switching.

An important issue of MAS model is the granularity of the simulation. Representing one molecule by one agent is not reasonable when biological systems (a simple cell) implies several millions of molecules. But most often, study of specific metabolic pathways allows to look at only part of the biological system. Moreover, we work on mitochondrion which is a small organelle within a cell and the Q cycle is only a part of the mitochondrial metabolism. Therefore, simulation of hundred thousands of agents can be considered representative of the behaviour of one mitochondrial compartment. But the question of the number of agents must be kept in mind especially when 3D representation is intended.

We have designed a generic `BioAgent` which can be used for modelling any enzymatic process. We also generated a set of rules for agent behaviour that are compatible with the modelling of any redox driven enzymatic complex.

This works is part of a more general project MitoScoP [1] which main goal is to model and to simulate the whole mitochondrial energetic metabolism.

Acknowledgements

We thank the ACI MitoScoP for different grants and very helpful scientific discussions.

References

[1] J.P. Mazat. Beurton-Aimar M. S. Prs N. Parisey. How to lock a model from a miscomposition of objects ? In *13th European Conference on Object-Oriented Programming*, june 2003.

[2] N. Cannata, F. Corradini, E. Merelli, A. Omicini, and A Ricci. An agent-oriented conceptual framework for biological systems simulation. In *International Workshops on Models and Methaphors from Biology to Bioinformatics Tools (NETTAB 2004)*, pages 167–180, Italy, September 5-7, 2004.

[3] Antony R Crofts. The cytochrome bc1 complex: function in the context of structure. *Annu Rev Physiol*, 66:689–733, 2004.

[4] O V Demin, I I Gorianin, B N Kholodenko, and H V Westerhoff. [Kinetic modeling of energy metabolism and generation of active forms of oxygen in hepatocyte mitochondria]. *Mol Biol (Mosk)*, 35(6):1095–1104, Nov 2001.

[5] Z.ZHANG L.HUANG V.M.SHULMEISTER Y.I.CHI K.K.KIM L.W.HUNG A.R.CROFTS E.A.BERRY and S.H.KIM. Electron transfer by domain movement in cytochrome bc1. *NATURE*, 392(677), 1998.

[6] T.J. Hutton. Making membranes in artificial chemistries. In *Proc. Workshop on Artificial Chemistry, at ALIFE9*, 2004 (to appear).

[7] Salim Khan, Ravi Makkena, Foster McGeary, Keith Decker, William Gillis, and Carl Schmidt. A multi-agent system for the quantitative simulation of biological networks. In *AAMAS '03: Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 385–392, New York, NY, USA, 2003. ACM Press.

[8] P. Mitchell. Coupling of phosphorylation to electron and hydrogen transfer by a chemiosmotic type of mechanism. *Nature*, 1961.

[9] P Mitchell. The protonmotive Q cycle: a general formulation. *FEBS Lett*, 59(2):137–139, Nov 1975.

[10] Gilbert N. and Troitzsch Klaus G. *Simulation for the Social Scientist.* Open University Press, 2005.

[11] DG Nicholls and Ferguson SJ. *Bioenergetics.* Academic Press, 2004.

[12] A. Zemirline et L. Marc P. Ballet. The biodyn language and simulator. application to an immune response and e.coli and phage interaction. *Journal of Biological Physics and Chemistry 4 (2004) 93-101*, September 2004.

[13] F. Zambonelli and A. Omicini. Challenges and research directions. *Agent-Oriented Software Engineering. Autonomous Agents and Multi-Agent Systems*, 9(3):253–283., 2004.

Interaction and emergent behaviour in heterogeneous groups of artificial agents

Sven Magg
Adaptive Systems Research Group
University of Hertfordshire
AL10 9AB Hatfield, UK
S.Magg@herts.ac.uk

René te Boekhorst
Adaptive Systems Research Group
University of Hertfordshire
AL10 9AB Hatfield, UK
R.TeBoekhorst@herts.ac.uk

Abstract

In robot experiments that try to investigate behaviour found in swarms of social insects, mostly homogeneous groups of agents were used. Task allocation within these groups was achieved by dynamic developmental processes. In contrast, this work accesses the dynamics in a heterogeneous group of simulated agents building structures in a simple world. Task diversification is predefined and the effects of group parameters like composition and agent's abilities are in the focus of interest. By analyzing the emerging patterns built by the heterogeneous group, it is shown that depending on the swarm configuration the resulting behaviour on group level can be robust to changes within the group. Different emerged patterns which are more sensible to parameter changes are also investigated and responsible effects explained.

1 Introduction

It is a commonly held view that more complex tasks require more complex robots, which contradicts the desire to keep them simple, robust and easy to maintain. To find new ways to solve this dilemma, some roboticists have turned to abstracting the essential features and dynamics of living systems. The field of collective robotics for example, in which particularly coordinated action is an issue, has been inspired by the biology of social insects.

An important mechanism in social insects to coordinate their actions, and which has been taken up by roboticists, is stigmergy. Stigmergy explains indirect task coordination without direct communication (Grassé 1959). Instead, communication is mediated by changes in the structure of the local environment that are reinforced by the behaviour of the agents. Since the behaviour of the agents in turn is triggered by the structure of the local environment, stigmergy is often associated with the notion of self-organisation. One robot application is the experiment by Maris and te Boekhorst (Maris 1996) in which robots (based on a Braitenberg architecture) pushed objects together.

Some proponents of self-organisation have suggested that essentially similar individuals can adopt different roles due to local interactions, hence playing down initial behavioural or morphological differences among agents. Also experiments with robots that exploit the self-organised effects of swarm behaviour typically start with homogeneous groups (Maris 1996; Labella 2004) .

However, in many animals, individuals with different morphological features do exist (see e.g. Detrain 1991) and in certain species form castes within a society. These differences allow particular castes to perform certain tasks better than others (e.g. defending the nest, foraging or processing food). Clearly, the relative importance of self-organisation on the one hand and initial morphological differences on the other hand is still a matter of debate (Bonabeau 1999). It is obviously an important issue for robotics to establish whether task allocation can be accomplished better by a combination of different types of agents than by a collection of identical agents solely based on self organization.

The experiments reported here were carried out as a first step to assess the dynamics within a predefined heterogeneous group of agents. More specifically, they address the question to what extend the composition of the colony (in terms of classes of agents) contributes to the complexity of pattern formation and the behaviour on colony level.

2 Experimental setup

An artificial world was designed in NetLogo. The 2-dimensional environment represents an area (either closed or toroidal) which contains "particles" and is populated with agents. Two types of agents can inhabit the world, Bull-dozer-type agents ("Dozers") and Grabber-type agents ("Grabbers").

Dozers move straight forward and push any single particle that is directly in front of them until it cannot be pushed any further (e.g. when it is pushed against the wall or other objects). If Dozers collide with an obstacle (wall, other agents or a pile of two or more particles) they change their direction randomly (left or right) by turning under an angle of 45 degrees. The agent may lose the particle that it pushes on the way with a probability proportional to the distance pushed, after which the agent turns 45 degrees in a random direction. Dozers also continuously deposit an artificial "pheromone". This pheromone triggers the behaviour of Grabbers. Grabbers can grab and carry particles (also when turning) instead of just pushing them ahead. In case they do not hold a particle, Grabbers move straight on, until they encounter an obstacle. If this obstacle is a wall or another agent, they turn 45 degrees in a random direction. If they run into a particle, and depending on the level of pheromone at that location, they grab it. The probability of grabbing increases with the local pheromone concentration. In case a Grabber already holds a particle, it moves straight ahead until it encounters an obstacle (wall, agent or box) after which its turns 45 degrees in a random direction. The probability of dropping the particle increases with decreasing pheromone level and the distance the particle was carried. In some experiments Grabbers also continuously drop pheromones with same intensity as Dozers.

In the experiments carried out so far, 65% of the area was filled with particles. All agents start in the middle of the world heading in a random direction and spread outwards after the start of an experiment. The total number of agents in the world always was 100 and the Dozer to Grabber ratio was permutated. All simulations were performed in a toroidal world as well as in a closed area and also with the ability of Grabbers to drop pheromones switched on and off. All experiments were run for 300,000 time steps. At each time step, all agents where consecutively activated. For each parameter setting, 10 runs were performed.

The structural patterns which emerged are nest-like structures that, to a varying degree, consist of "corridors" and "chambers" (Figure 1).

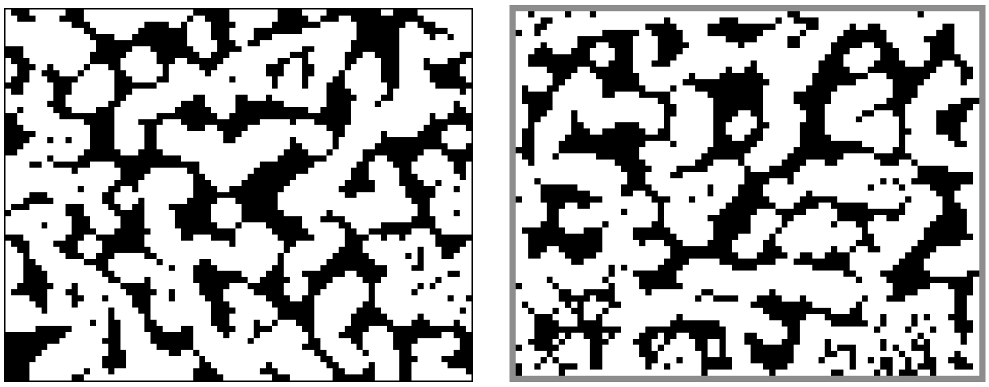


Figure 1*: Two example pictures. Each white pixel represents a particle an each black pixel represents an empty spot. Grey lines represent surrounding walls.*

The structure of the pattern was captured by the following measurements:

- **Number of chambers**. This measure is calculated using an algorithm which computes the distance from each empty spot to the nearest particle and in a second step searches for local maxima in the resulting 2-dimensional landscape of distance values at each pixel. If the picture is noisy (i.e. a lot of freestanding particles) the algorithm is unreliable and the pictures have to be manually reviewed.
- **Complexity.** Structural complexity is measured based on the *information dimension* (Baker 1990). The picture is covered by a grid and entropy was binned into three classes: a grid quadrant either consists completely of white pixels, black pixels or contains both. Therefore the entropy for a picture with a given grid size x is:

  $$E(x) = -\sum_{j=1}^{3} p_{x,j} \cdot \log_3(p_{x,j})$$

  where $p_{x,j}$ is the probability that a quadrant with side length x belongs to class j. The structural complexity is then given by the slope of the regression of E(x) for growing values of x starting with 2.
- **Number of piles.** A pile was defined as connected conglomeration of particles (using a Moore neighbourhood) with a cardinal number higher than 5. It was counted automatically by a breadth-first search algorithm consecutively marking neighbouring white pixels, incrementing the pile count and starting again with an unmarked white pixel.

3 Results of Homogenous Groups

Before talking about heterogeneous groups, we look at results from homogeneous groups of each type to show differences between the agent classes. Both agent classes show different behaviour on group level which is explained separately.

3.1 Dozers

The behaviour of the Dozers can be described as "keep areas free of particles". They push single particles to the walls delineating chambers. The resulting structure always consists of many narrow corridors (as can be seen in Figure 1) and many small chambers. This is due to the initial random distribution of particles; the maximum distance a particle can be pushed without colliding, is rather short. Once all movable particles are pushed against each other no further changes can occur. The final structure is static and for groups with size > 10 independent of group size. For simplicity, this pattern is from now on referred to as "Dozer-pattern". Examples can be seen in Figure 1. The number of chambers in this structure is generally more than 30 and the value of the complexity measure is around -0.25. If a single Dozer creates a small chamber, it often gets trapped in that chamber by locking itself in. However, with increasing numbers, this is prevented by the activity of other Dozers. Due to mutual avoidance movements in a group, Dozers change directions more often and therefore have an increased chance to find particles that can be pushed away. In this way they open up the area for their fellow Dozers to travel further and therefore prevent single or small groups of Dozers from getting trapped. As a result, with increasing group size also the chance to get the Dozer-pattern increases.

3.2 Grabbers

In contrast to Dozers, Grabbers generally move particles from areas with many agents to less densely populated areas. Because agents spend more time at obstacles due to their avoidance movements and therefore mark them strongly with pheromones, Grabbers tentatively move particles from larger piles to the centre of chambers.

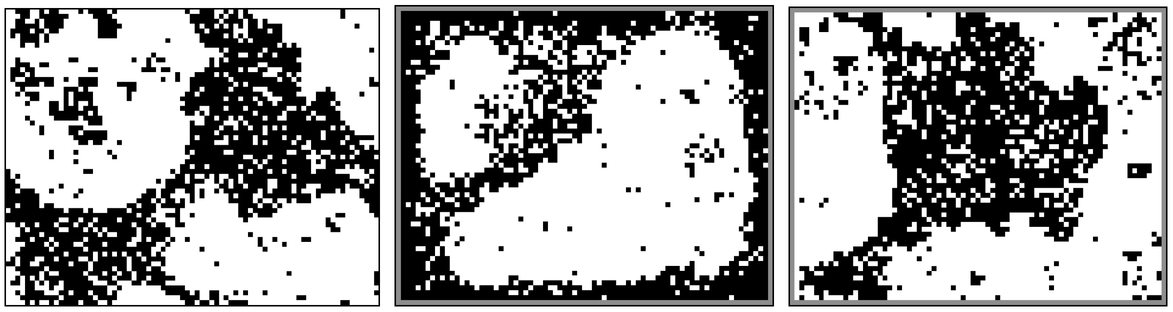


***Figure 2**: Sample patterns generated by a homogeneous group of 100 Grabbers (with ability to drop pheromones). On the left in a toroidal world, middle and right in a closed world.*

A homogeneous group of grabbers is only active if they are able to drop pheromones. Because the grabbing behaviour is dependent on the presence of pheromones, the Grabbers are inactive otherwise. As can be seen in Figure 2, the pattern generated by such a group is noisy, but on a closer look generally consists of a few large chambers connected by corridors (or from a different point of view a few big piles). This pattern changes dynamically over time but keeps its basic structure and is from now on referred to as "Grabber-pattern" in this paper. If the noise which strongly affects the measurements is removed, the number of chambers is generally less than 10 and the value of the complexity measure is around -0.05. The number of piles counted is normally less than 3.

4 Results of Heterogeneous Groups

The agents started to build piles of particles which generally lead to corridors and chambers, but their number and the complexity of the pattern depend strongly on the experimental parameters (number of agents, ratio Dozers/Grabbers, presence/absence of a border and Grabbers leaving pheromones or not). Dependent on these parameters, the results can be partitioned into three classes:

1. Grabbers don't drop pheromones
2. World has no borders and Grabbers drop pheromones
3. World has borders and Grabbers drop pheromones

In each experiment shown here, the ratio Dozers/Grabbers was varied from 100% to 0%.

4.1 Grabbers Not Dropping Pheromones

Starting with a Dozer-pattern for 0% Grabbers, the results quickly change to a more Grabber-like pattern when their percentage increases (chambers <10 and complexity value > -0.07). Compared to the Grabber-pattern described earlier, this pattern is noise free, i.e. contains no single standing particles and still has more chambers.

With further growth of the subgroup, their activity and therefore their influence on the resulting structure is decreasing, because less pheromones are dropped by the decreasing proportion of Dozers. In the extreme case of 100% Grabbers no particle is moved and the initially random world is not changed. Starting from 40% the result slowly converges to that of a Dozer-only group. As can be seen in Figure 3 and Figure 4 the presence or absence of walls has no impact.

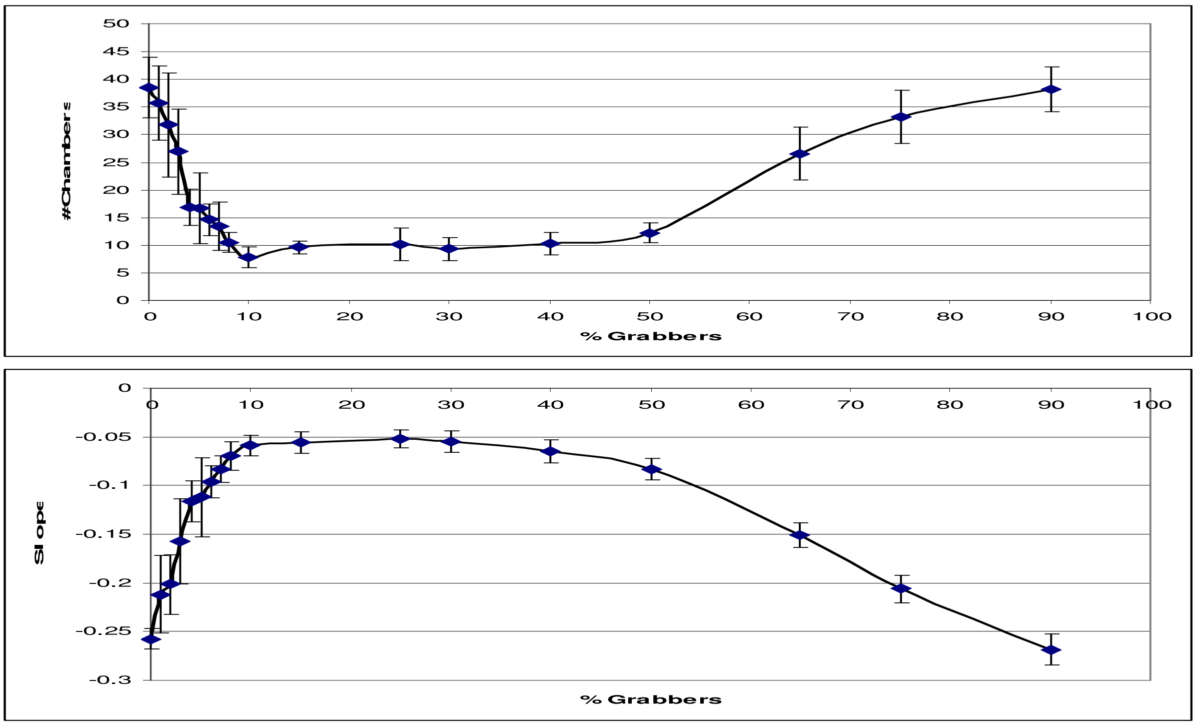


***Figure 3**: Chamber count (top) and complexity measure (bottom) for experiments with a heterogeneous group in a toroidal world and Grabbers not dropping pheromones. Error bars indicate the 95% confidence interval. 100% Grabbers is not plotted because they stay inactive and the result is the initial random particle distribution.*

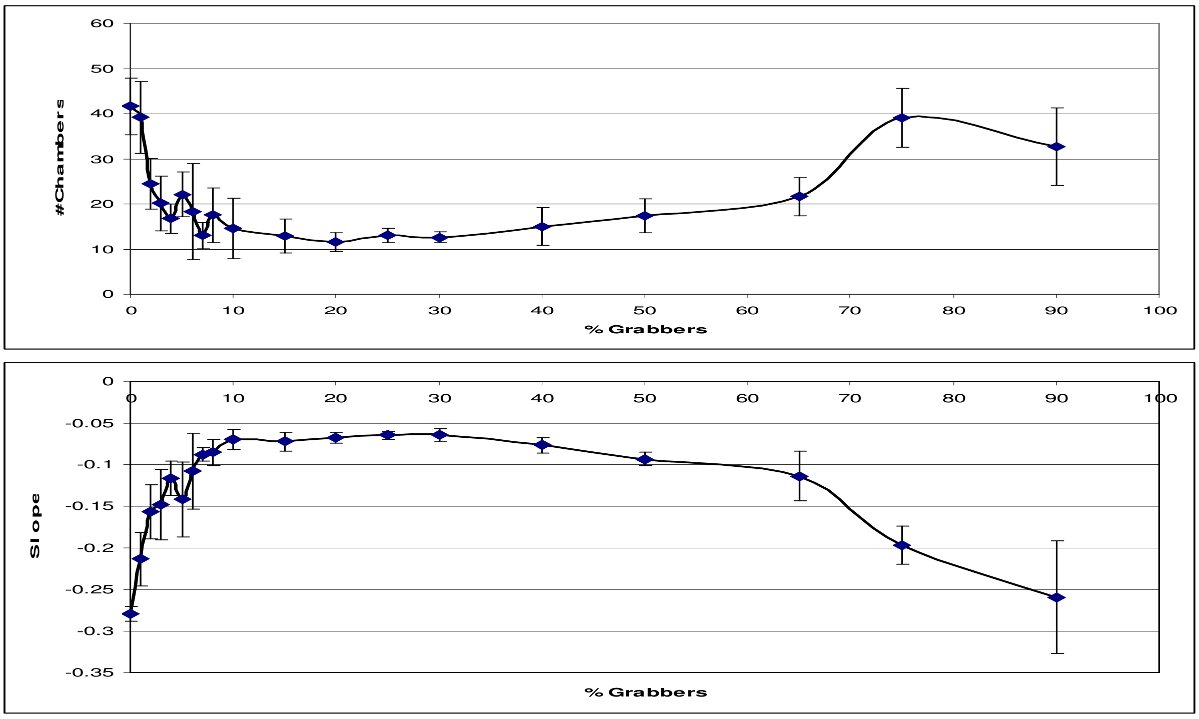


***Figure 4**: Chamber count (top) and complexity measure (bottom) for experiments with a heterogeneous group in a closed world and Grabbers not dropping pheromones. Error bars indicate the 95% confidence interval. 100% Grabbers is not plotted because they are inactive and the result is the initial random particle distribution.*

4.2 World has no borders and Grabbers drop pheromones

When Grabbers also drop pheromones, the overall pheromone level in the world stays the same because the number of pheromone-dropping agents stays constant over all experiments. Without borders and starting from a Dozer/Grabber ratio of 1, the structure gets noisier and converges to the pattern of a Grabber-only group. The increase in freestanding particles is due to the decreasing activity of Dozers that now pose a minority. With noise, the chamber counting algorithm gets unreliable which results in the increasing chamber count in Figure 5. Other than the graphs may suggest viewed sample patterns show no significant change in the basic structure. Changes in the graphs between 40% and 100% therefore have to be traced back to the increase in the number of free standing particles.

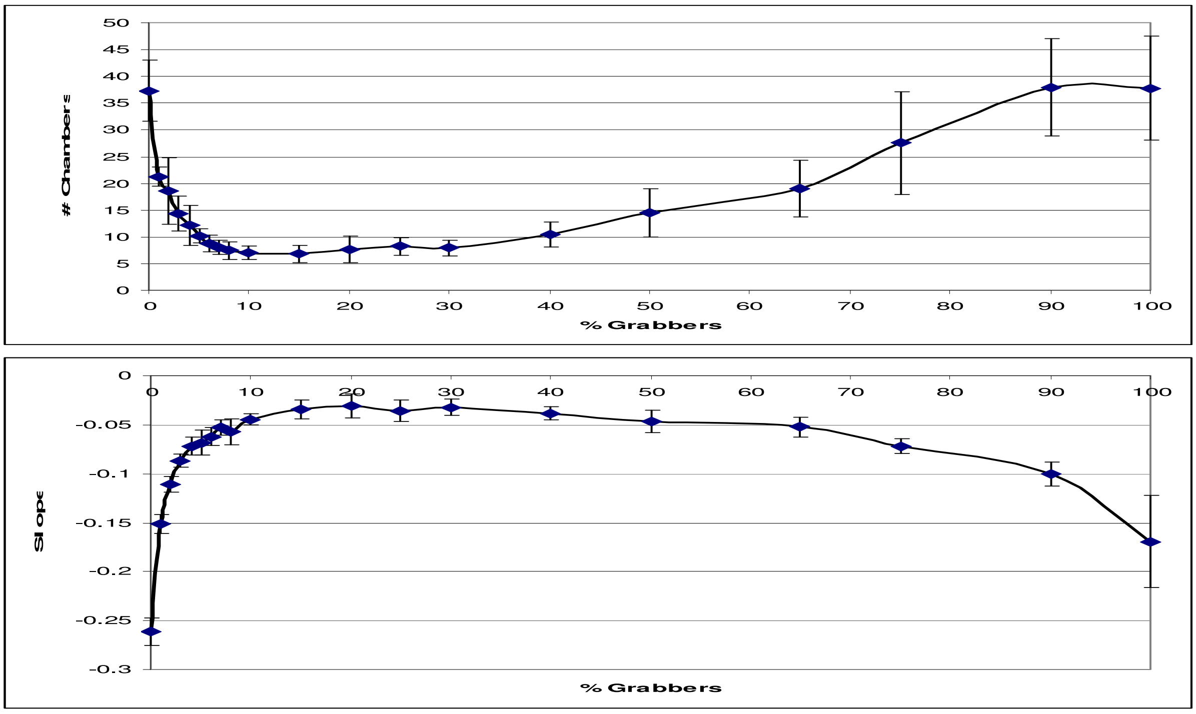


Figure 5: Chamber count (top) and complexity measure (bottom) for experiments with a heterogeneous group in a toroidal world and Grabbers dropping pheromones. Error bars indicate the 95% confidence interval.

4.3 World has borders and Grabbers drop pheromones

Other than in the case of Grabbers not dropping pheromones, the presence of walls in this setting has a big impact on the results of a group with Dozer/Grabber ratio less than 1. With increasing percentage of Grabbers, the chance to get one big pile in the middle of the world (see Figure 2, middle) also increases. The explanation is that Grabbers are likely to move boxes away from walls because the pheromone level near walls is usually higher than average. This is due to agents' propensity of wall following. Bumping against a wall and the consequent 45 degree turns make it likely that agents move along the wall. The pile counts (Figure 6) show, that the likelihood to build one pile increases when the ratio Dozers/Grabbers drops below 1, but decreases again when the percentage of Grabbers gets close to 100%. In the case of 90% only single particles can be temporally found at walls. The increasing chamber count (Figure 6) and the dropping complexity value for a ratio less than 1 can be again partly explained by the increase in noise. However, the amount of freestanding particles is less than in 4.2. This is due to the increased probability that areas along the wall get visited by Dozers travelling along them, thereby faster removing particles even when their number decreases.

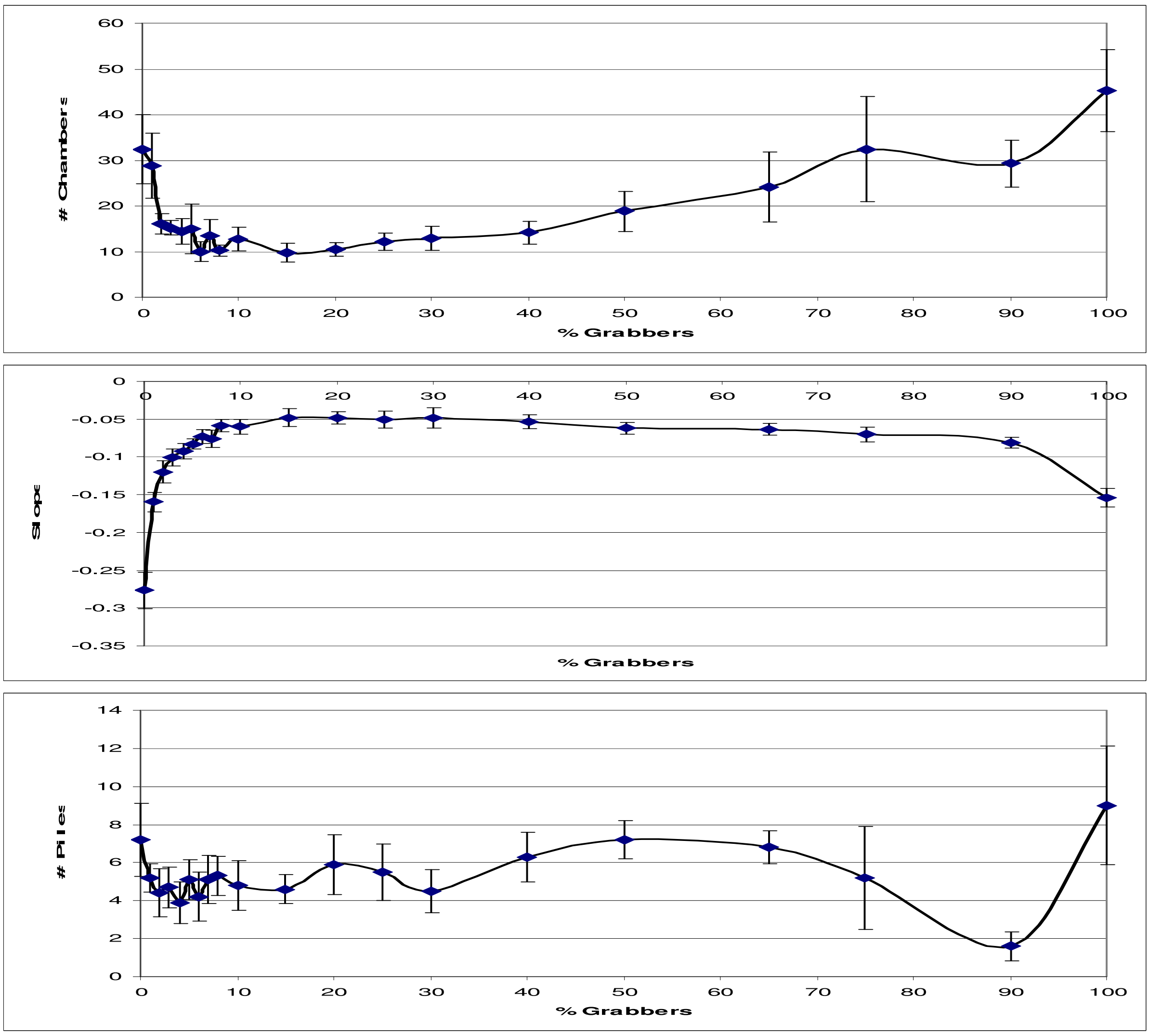


Figure 6*: Chamber count (top), complexity measure (middle) and pile count (bottom) for experiments with a heterogeneous group in a toroidal world and Grabbers dropping pheromones. Error bars indicate the 95% confidence interval.*

5 Discussion

Results shown give evidence, that the resulting structures built by a mixed group can't be easily derived from the behaviour of the individual subgroups and are in certain cases sensitive to small parameter permutations. Some observations on the combined behaviour of a heterogeneous group can be made:

- Dozers efficiently keep areas free from single particles, thus reducing noise in the structure. Particles dropped by Grabbers are pushed to a wall or conglomeration again.
- High pheromone levels produced by agents due to avoidance movements facilitate corridor building. A small dead end for example where many agents got stuck, gets opened up by Gabbers removing particles and Dozers pushing these out on their way back, hence building a corridor. Also bottlenecks and small chambers are generally enlarged by this mechanism.
- Because of the bigger surface area of larger conglomerations which is hit more often by Dozers, particles are tentatively moved from smaller to larger piles, thus removing smaller piles over time. A single conglomeration in the middle of a bigger chamber is often not affected because of less pheromone in its vicinity compared to the pheromone level at the surrounding chamber walls. This

is again due to the propensity of wall following, further increased by concave chamber walls in contrast to the convex surface of the pile.

Taking one step back and looking at the emerging patterns also shows features of heterogeneous groups which, if reproducible in other experimental settings, can be useful for the composition of artificial swarms. Regardless of the different parameters in the classes, the resulting patterns for groups with less than 25 Grabbers were always comparable (see Figure 3Figure 4Figure 5Figure 6). The starting point was always 100 Dozers, which led to the static structure described above. With increasing number of Grabbers the structure almost immediately changed as can be seen in Figure 7. Because Grabbers are able to change the static structure by taking particles away which Dozers are unable to move, the pattern continuously changes over time, but keeps its basic structure. This pattern can be compared to the Grabber-pattern, but contains almost no freestanding particles, because objects dropped by a Grabber are almost immediately pushed against a wall by a Dozer.

Together with the finding, that the resulting patterns for groups with 15 to 40 Grabbers are not significantly different, this gives evidence, that there is a "stable" group composition. In this stability region, small changes within the group have no impact on the resulting behaviour, hence making it robust to disturbances in the group. If it turns out that this also holds for groups of real robots it should strongly govern group composition to gain robustness and therefore performance.

Another finding was that for one parameter setting, the resulting pattern was always one big particle conglomeration in the middle (see 4.3). This behaviour was already shown in real robots (Maris 1996) and it is not known yet, why it only emerged in this particular setting. One reason could be that a majority of Dozers pushes particles faster to the wall than Grabbers can remove them. But even with a low number of Grabbers, certain areas at the wall can be cleared and kept open as Figure 7, middle and right show. These regions at the border also show a pheromone level above average, thereby increasing Grabber activity and the proportion of clear walls stays constant over time. More experiments are necessary to understand the reasons that yield to or inhibit the behaviour leading to a single pile.

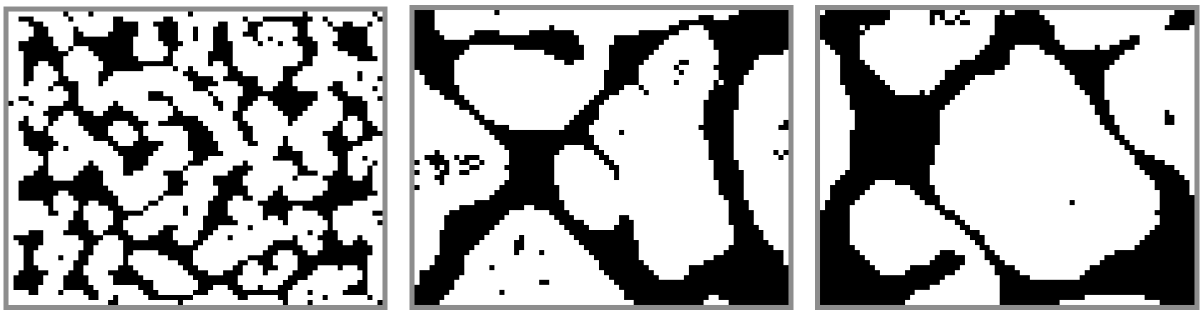


***Figure 7**: Pattern of a heterogeneous group with increasing percentage of Grabbers (0%, 10% and 25%) in an enclosed world*

The experiments performed so far show, that although the exact emergent structure is not predictable in this case, the general properties of the structure can be externally controlled by group parameters. They also show that easily overlooked interactions between agents-agents and agents-environment influence the resulting pattern and are partly accountable for the behaviour on group level. These effects have to be understood and to be taken into account when designing heterogeneous groups for specific tasks. More experiments have to be carried out to verify the findings with different group sizes and also access the influence of pheromone levels on the result. In which ways pheromone properties like evaporation and diffusion rate effect the results have not been analyzed, yet. It also has to be investigated if the findings are still valid for groups of robots in real world environments.

6 References

Baker, G. L. and Gollub, J. P. (1990). "Chaotic dynamics", Cambridge University Press.

Bonabeau, E., Theraulaz, G. and Deneubourg, J.-L. (1999). "Dominance Orders in Animal Societies: The Self-organization Hypothesis Revisited." Bulletin of Mathematical Biology 61(4): 727-758.

Detrain, C. and Pasteels, J. M. (1991). "Caste differences in behavioral thresholds as a basis for polyethism during food recruitment in the ant, *Pheidole pallidula* (Nyl.) (Hymenoptera: Myrmicinae)." Insect Behavior 4(2): 157 - 176.

Grassé, P.-P. (1959). "La reconstruction du nid et les coordinations interindividuelles chez *Bellicositermes natalensis* et *Cubitermes sp.* la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs." Insectes Sociaux 6(1): 41 - 80.

Labella, T. H., Dorigo, M. and Deneubourg, J.-L. (2004). "Self-Organised Task Allocation in a Group of Robots". in Proceedings of 7th International Symposium on Distributed Autonomous Robotic Systems (DARS04), Toulouse, France.

Maris, M. and te Boekhorst, R. (1996). "Exploiting Physical Constraints: Heap Formation through Behavioral Error in a Group of Robots". in Proceedings of IROS'96, IEEE/RSJ International Conference on Intelligent Robots and Systems, Senri Life Science Center, Osaka, Japan.

Emergent Sorting Patterns and Individual Differences of Randomly Moving Ant Like Agents

Alexander Scheidler, Daniel Merkle, Martin Middendorf

Parallel Computing and Complex Systems Group
University of Leipzig
04105 Leipzig, Germany
{scheidler,merkle,middendorf}@informatik.uni-leipzig.de

Abstract

Slight differences in the individual movement behaviour of a group of moving agents can lead to emergent patterns in the distribution of the agents. In this paper we study an emergent spatial sorting effect as it has been observed before in simulations of ants moving within their nest. In our study we use a cellular automaton model to investigate which elements of the movement behaviour can lead to spatial sorting. In the studies that have been done to simulate ants only one focal point - the centre of the nest - influences the movement behaviour of the ants. Here we consider also a scenario where more than one such focal point exist. This scenario is relevant for applications in robotics or organic computing where several service stations exists for a group of moving agents.

1 Introduction

Emergent patterns that occur when groups of simple agents move are obviously an interesting topic for biology (e.g., [1, 3, 7, 8]) and robotics (e.g., [2]). But it is also interesting for the design of modern computing systems that consist of moving objects or where parts of the system are embedded into moving objects. Such systems are addressed by the organic computing initiative where the aim is to construct computing systems that can adapt to their environment or the user and have so called self-x properties, e.g., they should for example be self-configurable, self-optimizing, and self-servicing (e.g., [4]). One idea of organic computing is to use principles of self-organization as they are found in natural systems for the design of computing systems. Since self-organized systems can show emergent effects it is important to understand when and what type of emergent effects can occur. Typically, in organic computing systems the members of the system can adapt to environmental conditions. Hence, even if the members are all equal in principal they will show slightly different behaviour due to the individual adaptations. Therefore, it is interesting to investigate what type of emergent effects might occur due to such slight differences in individuals behaviour. In this context, ant colonies are one particularly interesting natural system because they consist of relatively simple individuals and can show complex and emergent behaviour.

In this paper we study the emergent sorting behaviour of simple ant like moving agents (see [6]).Sorting here means that agents with different behaviour can be found most often in different parts of the nest area (or movement area). The starting point of our investigation is the study of Sendova-Franks and Lent [6] where they simulate the movement behaviour of real ants in their nest. Using four different models of moving behaviour it was shown that sorting occurs in all models. In all models the moving behaviour of the ants depends on a parameter μ. The strongest sorting effect occurred for the centripetal ant model where parameter μ determines the strength of an attraction force towards the nest centre. For the other three models without attraction force parameter μ determines the maximal turning angle of an ant during movement. Ants with a small turning angle tend to keep to the wall once they have collided with it. Thus it was concluded in [6] that the colony centre or the wall can play the role of a pivot (or beacon) which appears to be necessary for the sorting.

First, we investigate the occurrence of high concentrations of agents in the centre of the nest area as observed before in the simulations of [6] in more detail. It is shown here that the agents abundance in the centre is so high that they get stuck by each other. Therefore, most agents in the centre do not move most of the time. In order to investigate whether this relatively high concentrations of agents can also occur when the agents still have enough space to move we changed the movement behaviour slightly. Secondly, we introduce some changes to the movement model to obtain a behaviour that is more realistic for robotic applications and to find out which behavioural differences can lead to an emergent sorting behaviour. Thirdly, we investigate a movement model with attraction force for the case that there is more than one centre of attraction. It is shown that there exist an interesting emergent effect so that different centres can attract ants with various μ value differently. This scenario is interesting for organic computing, e.g., when there are several service stations for a group of moving agents (which are members of an organic computing system).

In Section 2 the continuous ant movement model of [6] is described. The cellular automaton model is introduced in Section 3. The experiments are described in Section 4 and the results are presented in Section 5. Conclusions are given in Section 6.

2 Continuous Model

A continuous model for ants movement in a nest was introduce in [6]. In this model the simulated nest is a rectangle with dimensions $30 \times 20\text{mm}^2$ (similar to the dimension of real nests that are used in biological studies [9, 10]). The number of ants in the population is $n = 40$. The shape of an ant is modelled as a disc with radius $\rho = 1$ mm. The centre of the disc (x_i, y_i) represents the position of ant i. Each ant has an actual direction of movement α_i, which is measured as the angle relative to the lower border of the nest. The point at position $(x_i + \rho \cos \alpha_i, y_i + \rho \sin \alpha_i)$ models the centre of the ants head. From the head every ant can sense obstacles in a range of $\sigma = 0.6$ mm, called sensing distance (see Figure 1 a). Ant i collides with ant j if it is within its sensing range, i.e., when the distance between the centre of the head of ant i and the centre of the body of ant j is smaller than $\sigma + \rho$ (Figure 1 b). Similarly, an ant has collided with the nest wall when the euclidian distance between the centre of its head and the wall is less than the sensing distance.

In every time step each ant changes its actual movement direction by $\alpha_i = \alpha_i + \theta_i$ (the calculation of the turning angle θ_i differs for various movement models as described below). An ant changes its position only when it is unobstructed, i.e., no nestmate or nest wall is within its sensing range. If so, the ant moves distance $\delta = 0.3$ mm in direction α, i.e., $x_i \leftarrow x_i + \delta \cos \alpha_i$ and $y_i \leftarrow y_i + \delta \sin \alpha_i$. Behavioural differences between individuals are modelled so that each individual i has a parameter μ_i that influences its moving behaviour. Value $\mu_i = i/(n+1)$ for ant $i \in 1, \ldots, n$ was used for the experiments in [6]. The following two models of moving behaviour have been studied in [6].

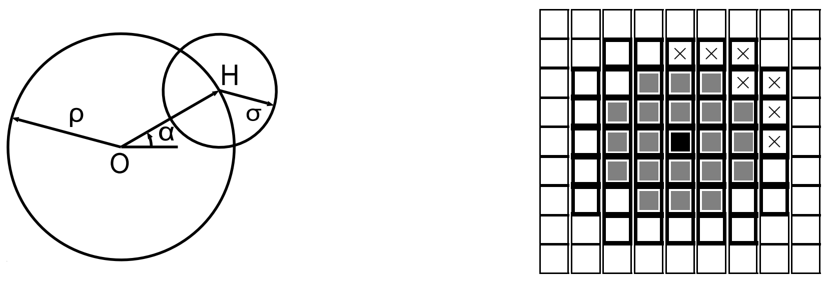


Figure 1: (a) Every ant in the continuous model is modelled as a disc; ρ - radius; O - centre of the body (x, y); α - direction of movement; H - centre of the head; σ - sensing distance. (b) Shape of ant in the cellular automaton model; the black cell represents the centre of the ant; the fat black cells are in the neighborhood N; the gray cells belong to the body of ant N_B; cells with cross belong to $N^{(1,1)}$, only if these cells are free the ant can move in direction (1,1)

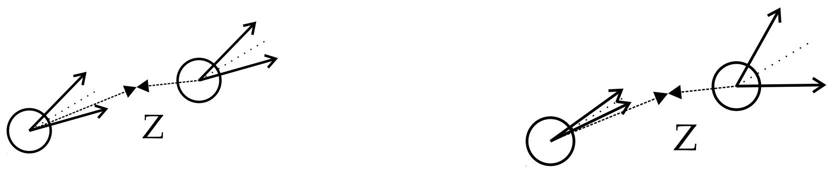


Figure 2: Effect of different internal parameter μ_i on the turning behavior in the Centripetal Ant Model; Z is the centre of the nest; (a) for large μ_i there is only a slight difference between moving from or to the centre; (b) for small μ_i the turning angle becomes significantly smaller the larger the angle between actual moving direction and the vector to the centre is

Avoiding Ant Model. In this model the ants do a correlated random walk. If unobstructed, i.e., an ant does not collide with a nestmate or the nest wall, the movement is as follows. The turning angle θ_i is chosen randomly according to a uniform distribution between $-\Theta_i$ and Θ_i. The maximal turning angle Θ_i is different for all ants and depends on their individual parameter μ_i: $\Theta_i = (1 - \mu_i)\Theta^0 + \mu_i\Theta^1$. In [6] the standard values were $\Theta^0 = 60$ and $\Theta^1 = 15$. If the nest wall or a nestmate is in the sensing range, the ant will not move but only change its moving angle. In this case it avoids this obstacle explicitly by turning into one direction until it can move again. To define the turning direction assume that ant i collides with ant j. The sign of the scalar product between the vector that is perpendicular to the vector of the moving direction of ant i and the vector from the centre of ant i to the centre of ant j determines the direction of turning: $\theta_i \leftarrow \text{sign}((-\sin\alpha_i, \cos\alpha_i) \cdot (x_j - x_i, y_j - y_i))U(0, \Theta_i)$. A collision with the nest wall is handled analogously.

Centripetal Ant Model. Its very likely that ants have the possibility to detect gradients in gas (CO_2) or pheromone concentrations [5]. Since the concentration of the gas is maximal in the centre region of the nest where the brood is located [12], this could give the ant a chance to estimate the direction to the centre. In the Centripetal Ant Model this is used to establish a attraction force towards the centre of the colony. This attraction is different for different ants and depends on their internal parameter μ_i. For the calculation of the moving behavior a modified model from the clinotaxis model from [11] is used: $\theta_i \leftarrow p_u\chi_i + p_b\tau_i \cdot (1 - \cos(\phi_i))/2$ where ϕ_i is the angle between the actual moving direction α_i and the vector towards the centre of the nest. The values of p_u and p_b are randomly chosen from $\{-1, 1\}$ and they determine the direction of turning. The turning behaviour depends on ϕ_i and the larger this angle the more the ant will turn. The parameter χ_i and τ_i depend on the internal parameter μ_i of the ant: $\chi_i \leftarrow (1-\mu_i)\chi^0 + \mu_i\chi^1$ and $\tau_i \leftarrow (1-\mu_i)\tau^0 + \mu_i\tau^1$ with $\chi^0 = 0°$, $\chi^1 = 15°$, $\tau^0 = 30°$, and $\tau^1 = 0°$. Agents with larger μ_i will not be that much affected by their ϕ_i as ant with small μ_i (see Figure 2). Therefore, for the ants with small μ_i the attraction to the colony centre is larger than for ants with large μ_i.

3 Cellular Automata Model

For the experiments in this paper a probabilistic cellular automaton model was designed which is suitable to approximate the behaviour of the continuous model of [6]. The cellular automaton has an array R of cells and a neighbourhood $N := \{(x,y) \mid x^2+y^2 \leq 13\}$ where x,y are integers, i.e., cell (x_0,y_0) has all cells (x_0+x, y_0+y) with $x^2+y^2 \leq 13$ as neighbours. For most experiments $R = \{1,\ldots,75\} \times \{1,\ldots,50\}$ was used. Then each cell corresponds to a $0.4 \times 0.4\text{mm}^2$ square of the continuous model. The body of an ant in the cellular automaton consists of 21 cells arranged in a circle. Formally, an ant at position (x_0,y_0) occupies all cells $(x_0,y_0)+n, n \in N_B$, where $N_B := \{(x,y) \mid x^2+y^2 \leq 5\}$ where x,y are integers (see Figure 1 b). Each ant i has an internal parameter $0 \leq \mu_i \leq 1$ and an actual direction of moving $0 \leq \alpha_i < 2\pi$. For each ant the probabilities to move to one of the cells of the Moore neighbourhood $n \in N_M = \{(-1,-1),\ldots,(1,1)\}$ are calculated. In order to move in a given direction $n \in N_M$ all cells of the new place must be free, i.e., the cells at $N^n = \{a+n | a \in N_B \wedge (a+n) \notin N_B\}$ must be empty (and within the R).

In the cellular automaton model at each time steps the ants move in random order so that each ant moves only one cell per time step. Since an ant can move only to discrete positions it might not be possible for an ant to move exactly in direction α. Therefore, an ant has a probabilistic movement behaviour where α is its expected direction. Similarly, the expected velocity is 0.3 mm/s = 0.75 cells/time step on average when considering a free run of an ant with no obstacles. Consider the case $0 \leq \alpha \leq \frac{\pi}{4}$. Let p_α and q_α with $p_\alpha + q_\alpha \leq 1$ be the probabilities to move in directions (1,1), respectively (1,0). Then the expected value for the movement direction is

$$E(\overrightarrow{v}) = p_\alpha \begin{pmatrix}1\\1\end{pmatrix} + q_\alpha \begin{pmatrix}1\\0\end{pmatrix} + (1-p_\alpha-q_\alpha)\begin{pmatrix}0\\0\end{pmatrix} = \begin{pmatrix}p_\alpha+q_\alpha\\p_\alpha\end{pmatrix} \tag{1}$$

and the expected velocity is

$$E(v) = p_\alpha \cdot \sqrt{2} \cdot 0.4 \ + q_\alpha \cdot 0.4 \ + (1-p_\alpha-q_\alpha)\cdot 0 \ . \tag{2}$$

Solving these equations leads to $q_\alpha = 3/4 - p_\alpha \cdot \sqrt{2}$ and $p_\alpha = 3/(4\cdot(1/\tan\alpha - 1 + \sqrt{2}))$. The general case of $0 \leq \alpha \leq 2\pi$ can be handled similarly.

In addition to the movement models of [6] that are used for the cellular automaton simulations we introduce three new movement variants here. Note that, if not stated otherwise, the turning behaviour of the ants is realized according to the Centripetal Ant Model.

Model with Repulsive Behaviour (Repulsive Model). As it is shown in Section 5, in the Centripetal Ant Model the ants form a cluster with many non-moving ants in the centre. To avoid this artifact the Centripetal Ant Model is slightly changed here by modifying the ants behaviour in the case of a collision with a nestmate or the nest wall. In this case the ant turns according the to Avoiding Ant Model. Otherwise the moving behavior remains as in Centripetal Ant Model. Therefore in the Repulsive Model the turning behaviour of the ant is different for different situations.

Model with speed differences (Speed Model). In this model the ants have different velocities. They are equidistantly distributed between 0 and 1, i.e., $\nu_i = \nu_0 + (i-1)(1-\nu_0)/(n-1)$

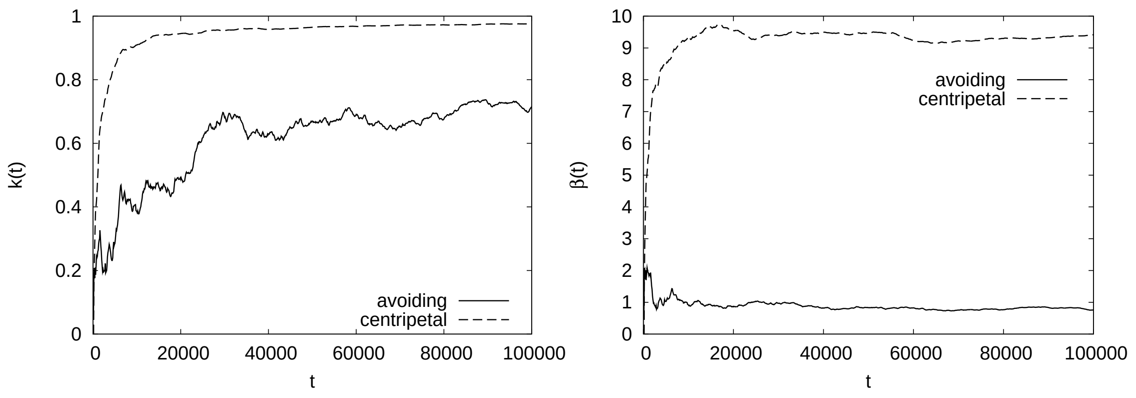


Figure 3: Simulation with stochastic cellular automaton; correlation coefficient $k(t)$ (left) and slope $\beta(t)$ (right) for the Avoiding Ant Model and the Centripetal Ant Model for different time steps

with $i \in \{1 \ldots n\}$ and $0 < \nu_0 < 1$. Note, that ant i moves ν_i-times as fast as ant n. The position of the ants is now updated according to $x_i \leftarrow x_i + \nu_i \delta \cos \alpha_i$ and $y_i \leftarrow y_i + \nu_i \delta \sin \alpha_i$. The different velocities are realised in the stochastic cellular automaton such that ant i has expected velocity $\nu_i \cdot 0.3$mm/s. (Recall that 0.3mm/s is the speed in the continuous model). The movement and turning behavior are the same for all ants and do not depend on their internal parameter μ. All ants move and turn like an ant in the Repulsive Model with $\mu_i = 0$.

Model with activity differences (Activity Model). The Activity Model is similar to the Speed Model. The difference is, that the ants do not only have different velocities, but also different turning behaviour. Similar to the velocities, the turning angle is scaled by ν_i. Formally, if ant i can move its turning angle is determined according to $\alpha_i \leftarrow \alpha_i + \nu_i \cdot \theta_i$, with θ_i calculated as in the Centripetal Ant Model. If the ant is obstructed, the turning behavior is defined as in the Avoiding Ant Model (see Section 2).

4 Experiments

If not stated otherwise all experiments were done over 100000 time steps and with 40 ants. As colony centre the point $Z = (35, 25)$ was chosen. To compare the results of the cellular automaton model with the results of the continuous model the distance of ant i to the colony centre is computed as $r_i(t) = d((x_i * 0.4 - 0.2, y_i * 0.4 - 0.2), (15, 10))$ where (x_i, y_i) is the centre of ant i in time step t. The distance r_i of ant i to the colony centre is measured every 100 time steps. For a given time step t the mean distance $r_i^{\o}(t)$ of ant i to the centre is the average over all measured distances $r_i(t)$ for $t = 0, 100, \ldots, c \cdot 100$ such that $c \cdot 100 \le t < (c+1) \cdot 100$. Let $r_i^{\o} = r_i^{\o}(100000)$ the average distance measured over all time steps.

Similar as in [6] Pearson's correlation coefficient was used to measure for the correlation of parameter μ_i and the mean distance of an ant to the nest centre. A high value (≈ 1) for $k(t)$ indicates a strong correlation. As a second measure the slope of the linear relationship between the mean ant distance to the colony centre and the internal parameter μ_i was measured. Given the mean distances $r_i^{\o}(t)$ for all ants $i = 1, \ldots n$, the linear regression determines values $\alpha(t)$ and $\beta(t)$ such that the sum $\sum_{i=1 \ldots n} (r_i^{\o}(t) - (\alpha(t) + \beta(t)\mu_i))^2$ is minimized. The slope $\beta(t)$ can be seen as a measure for the degree of sortedness of the ants.

To depict the spatial distribution of the ants a similar strategy was used as proposed in [6]. The nest was divided into $15 \times 10 = 150$ squares and the ants were divided into 5 groups depending on their internal parameter μ_i : $[0.0, 0.2)$, $[0.2, 0.4)$, $[0.4, 0.6)$, $[0.6, 0.8)$, $[0.8, 1.0)$. For each of

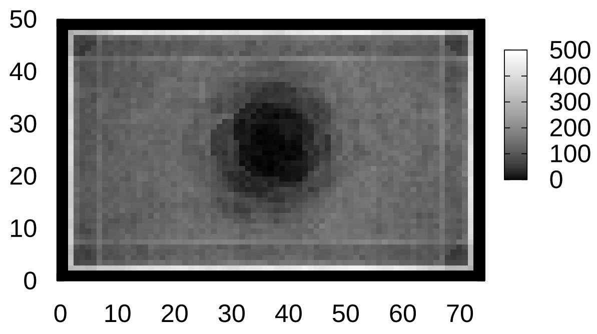
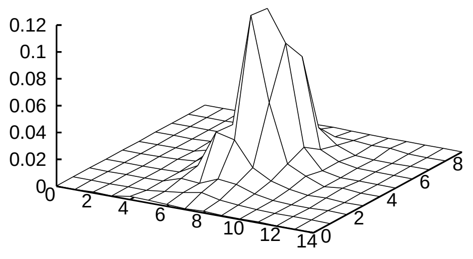


Figure 4: Centripetal Ant Model: frequency of how often an ant enters a cell within 100000 time steps


(a)

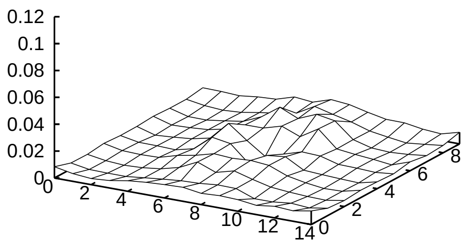

(b)

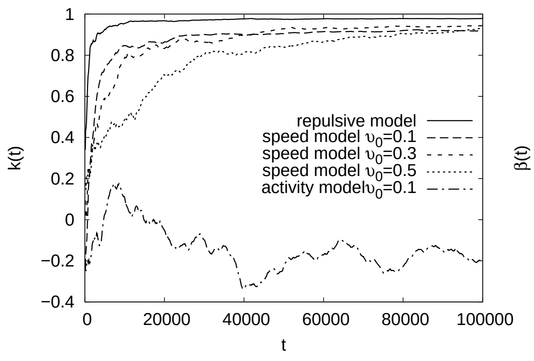
Figure 5: Spatial distribution of ants in classes $\mu \in [0, 0.2)$ (left) and $\mu \in [0.4, 0.6)$ (right)

these intervals the number of ants in every square was counted and divided by the total number of ants in the group. To investigate how active the ants are in different areas of the nest is was counted for every cell how often an ant enters that cell (measured over all time steps).

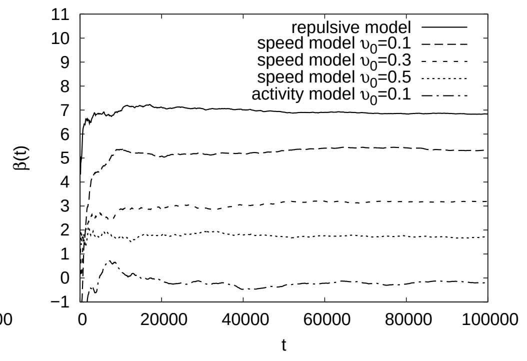
5 Results

Before the main results are presented it has to been shown that the cellular automaton model is a good approximation of the continuous model. Therefore some experiments of [6] have been repeated with the cellular automaton model using the Avoiding Ant Model and the Centripetal Ant Model. The left part of Figure 3 shows the correlation of the parameter μ_i and the distance of ant i to the nest centre. The change of the slope values $\beta(t)$ over time are shown in the right part of the figure. Both figures indicate a sorting behavior (much smaller for the Avoiding Ant Model). The relative frequency distribution of the ants is presented in Figure 5. Given is the relative frequency for ants with $\mu_i \in [0, 0.2)$ and $\mu_i \in [0.4, 0.6)$ in the cellular automaton model. All these results are in strong correspondence to the results presented in [6] (cmp. the corresponding figures in [6]).

Compared to the Avoiding Ant Model the Centripetal Model shows the stronger sorting effect. In order to investigate Centripetal Model in more detail it was measured for every cell how often an ant enters that cell. The results are shown in Figure 4. It can be seen that there is a cluster of non-moving ants in the center of the nest. This effect has not been observed before in [6] and

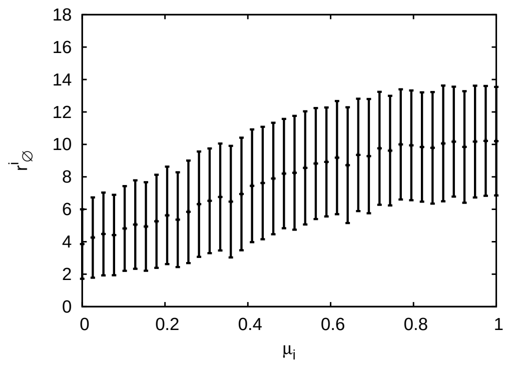


(a)



(b)

Figure 6: Changes of correlation coefficient $k(t)$ (left) and slope $\beta(t)$ (right) for Repulsive Model, Speed Model, and Activity Model



(a)

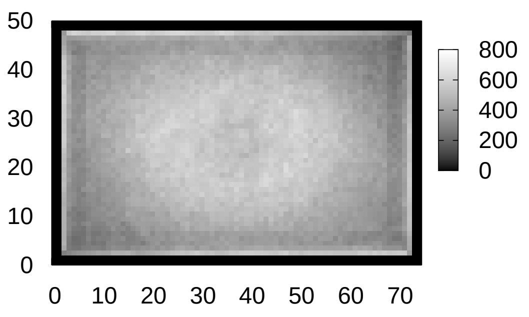

(b)

Figure 7: Repulsive Model: average distance to nest centre $r_i^{\varnothing}$ (left), and number of time steps where an agent enters a cell (right)

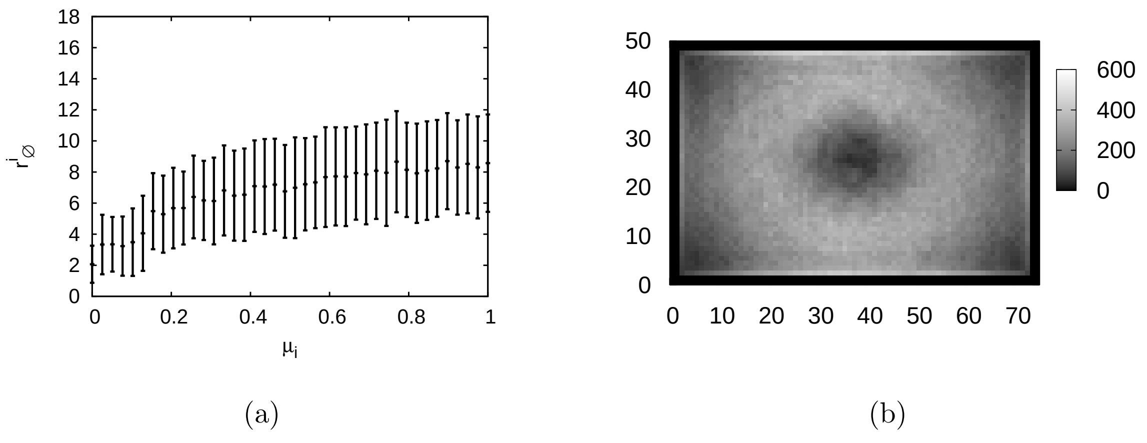
was one motivation to introduce new movement models where the ants try to avoid a situation where they get stuck.

5.1 Emergent Sorting

Figure 6 compares the new movement models introduced in Section 3 with respect to the correlation coefficient $k(t)$ and changes of the slope $\beta(t)$ of the regression function. The figure shows that in the Repulsive Model and in the Speed Model the ants shows a clear sorting behaviour. In the Activity Model there is no clear indication for a ant sorting.

The motivation to introduce the Repulsive Model was to avoid that the ants get stuck in the centre of the nest. Figure 7 (a) shows that ants with $\mu \approx 0$ have an average distance of approximately 4 from the centre whereas ants with $\mu \approx 1$ have an average distance of approximately 10. Thus there is a clear sorting behaviour. It can be seen in Figure 7 (b) that the ants in the centre have a high movement activity and do not get stuck.

The ants in the Speed Model show a strong sorting behaviour where slower ants can be found more in the centre of the nest. For large relative differences in movement speed ($\nu_0 = 0.1$) the sorting behaviour is stronger than for smaller differences ($\nu_0 = 0.3$) (compare Figure 8 (a) and (c)). It can also be seen that the ants in the centre (and in the corners) move much less than ants



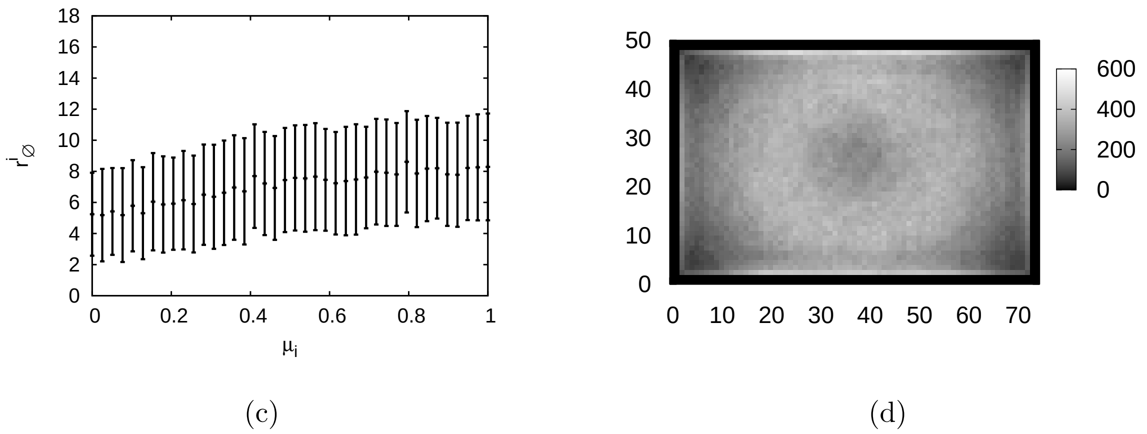


Figure 8: Speed Model: average distance to nest centre $r_i^\varnothing$ (left), and number of time steps where an agent enters a cell (right); movement speed $\nu_0 = 0.1$ (top) and $\nu_0 = 0.3$ (bottom)

in other parts of the nest. This effect becomes stronger with higher relative speed differences.

In the Activity Model no significant sorting behaviour can be observed. There are slight differences in the movement activity within the nest area. But these differences can be explained by the rectangular shape of the nest area. Ants can not move so easily into the corner of the nest area.

5.2 Complex Environments

For applications in robotics and organic computing complex environments with more than one focal point for the movement behaviour are interesting. An example are moving agents which have several service stations they can visit. Therefore, we investigated a much larger environment of 600×400 cells with two centres (located at $(150, 200)$ and $(450, 200)$. Simulations were done with in the Speed Model with 200 ants. The area is divided vertically at position $d \times 600$ such that in the left (resp. right) area the turning behaviour of the ants is influenced by the left (resp. right) centre, i.e., the ants tend to turn toward the corresponding centre. Note, that for $d = 0.25$ the line dividing both areas passes exactly the left nest centre. Again, the ants are divided in 5 classes according their μ-values. For both areas and all classes the number of ants in the left and right area are counted for $d \in \{0.25, 0.30, \ldots, 0.50\}$. The number of ants in time steps 0, 2000, and 50000 (results are averaged over 100 runs) are given in Figure 9. A clear differentiation of the five classes is evolving over time. At the beginning the number of ants

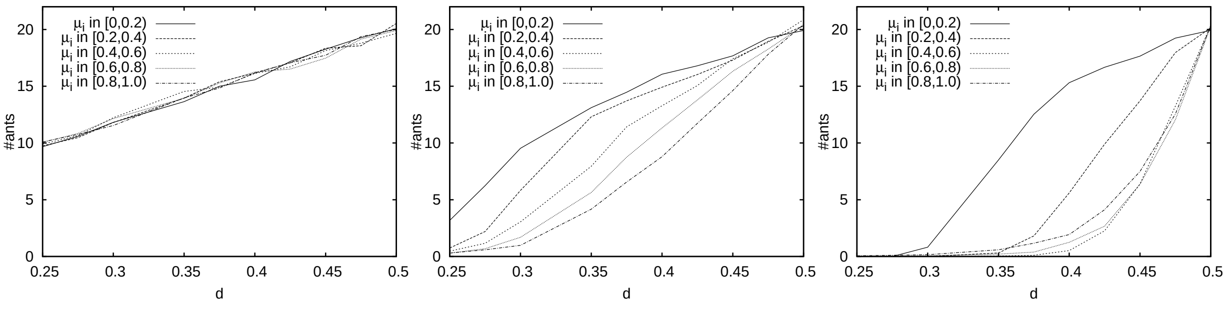


Figure 9: Two nest centres; average number of ants for different classes $\mu_i \in [0.0, 0.2)$, $[0.2, 0.4)$, $[0.4, 0.6)$, $[0.6, 0.8)$, $[0.8, 1.0)$ in the smaller of the two areas at time step 0 (left), 2000 (middle), and 50000 (right); results are averaged over 100 test runs

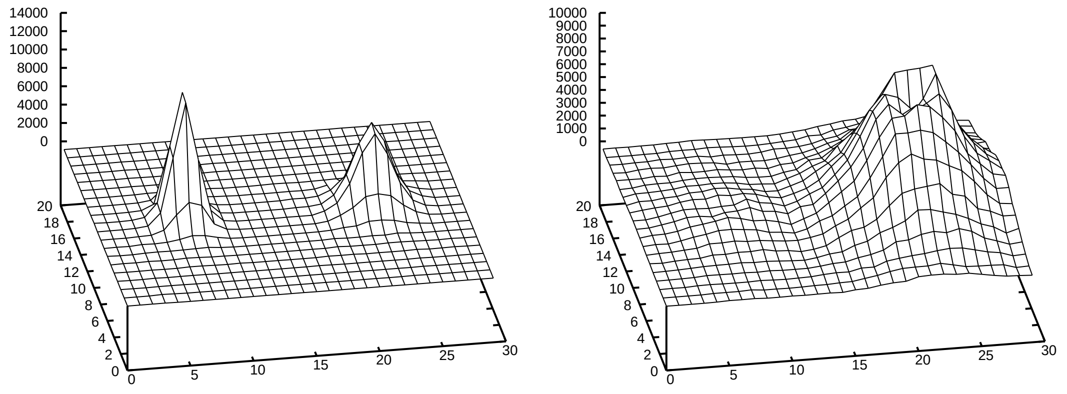


Figure 10: Two nest centres; $d = 0.4$; $\mu_i \in [0, 0.2)$ (left) and $\mu_i \in [0.8, 1]$ (right); for both influence areas it is counted how often an agent enters a cell (measured over 50000 time steps)

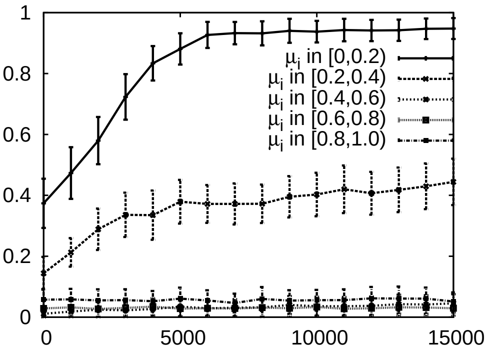


Figure 11: Activation of the nest centre in the larger area after $t \in \{0, 1000, \ldots, 15000\}$ simulation steps; $d = 0.4$; depicted is the fraction of agents for different classes $\mu_i \in [0.0, 0.2), [0.2, 0.4), [0.4, 0.6), [0.6, 0.8), [0.8, 1.0)$ that are in the smaller of areas after 50000 simulation steps

are equally distributed among the 5 classes for different values of d. Over time the faster ants (large μ_i values) occur more often than the slow ants in the larger area. Figure 10 compares the movement activity of ants with $\mu_i \in [0, 0.2)$ and ants with $\mu_i \in [0.8, 1)$ in different parts of the nest area for $d = 0.4$. It can be seen that the slow agents occur next to both centres whereas the fast agents occur mainly in an area around the centre with the larger influence region. This results show that moving agents with slightly different moving behaviour can have very different spatial distributions in areas with several several service stations.

We also interested in dynamic scenario where, e.g., the service stations for the agents do not occur at the same time. In the corresponding experiment it was assumed that the centre in the larger area became active several time steps later than the centre in the small region. The results are given in Figure 11 for the case that the centre in the larger area became active at time step $t \in \{0, 1000, \ldots, 15000\}$. Depicted is the fraction of agents in each of the classes $\mu_i \in [0.0, 0.2), [0.2, 0.4), [0.4, 0.6), [0.6, 0.8), [0.8, 1.0)$ that are in the smaller area after 50000 simulation steps. It can be seen, that the slow agents with $\mu_i \in [0.0, 0.2)$ are much more concentrated within the small area if the centre in the other area becomes active late (more than 90% of these agents appear in the smaller area if the service station appeared after >6000 simulation steps). The reason for this is that most of the slower agents are fast enough concentrate around the centre in the small region >6000 during the first. After that time they will leave the small area only with a very small probability. On the other hand it can be seen that a large fraction of the faster agents ($\mu_i > 0.4$) can always be found in the larger area regardless when the second centre was added. This mechanism maybe used to implement a controlled separation process of agents with different properties in organic computing systems.

6 Conclusions

In this paper we have investigated emergent spatial sorting patterns for groups of randomly moving ant like agents. Based on the continuous models introduced in [6] we used stochastic cellular automaton models to reexamine and extend those models. We have introduced three new movement models, the Repulsive Model, the Speed Model, and the Activity Model which are interesting for agent movement. It was shown that an artifact of a cluster of non-moving ants that emerges in one of the existing movement models is not the reason for a strong sorting behaviour. Moreover, the influence of different aspects of the movement behaviour on the emergent spatial sorting has been studied by simulations. A new type of scenario which has more than one centres has also been studied. It was argued that such scenarios are relevant for applications in robotics and organic computing where the centres can be seen as service points for the agents. It was shown that the relative size of the influence area of the service points leads to an emerging effect that the spatial distribution of agents might differ strongly with only slightly different moving behaviour. A dynamic scenario where different times spans between the activation times of both service stations were considered. It was shown that the length of this time span has a significant influence on the distribution pattern of the agents and the type of influence is different for different moving behaviours.

References

[1] J.-L. Deneubourg, A. Lioni, and C. Detrain. Dynamics of Aggregation and Emergence of Cooperation. Biol. Bull. 202: 262-267, 2002.

[2] J.-L. Deneubourg, S. Goss, N. Franks, A.B. Sendova-Franks, C. Detrain, and L. Chretien. The dynamics of collective sorting: Robot-like ants and ant-like robots. In *Proc. of the 1st Int. Conf on Simulation of Adaptive Behavior*, 356–363, 1991.

[3] S. Depickère, D. Fresneau, J.-L. Deneubourg. A basis for spatial and social patterns in ant species: dynamics and mechanisms of aggregation. Journal of Insect Behavior, 17(1): 81-97, 2004.

[4] GI: Organic Computing / VDE, ITG, GI - Positionspapier. 2003, online: http://www.betriebssysteme.org/Betriebssysteme/FutureTrends/oc-positionspapier.pdf

[5] G. Nicolas and D. Sillans, Immediate and latent effects of carbon dioxide on insects. Annual Review of Entomology, 34:97–116, 1989.

[6] A.B. Sendova-Franks and J.V. Lent. Random walk models of worker sorting in ant colonies. Journal of Theoretical Biology, 217:255-274, 2002.

[7] R.V. Sole, E. Bonabeau, J. Delgado, P. Fernandez, J. Marin. Pattern formation and optimization in army ant raids. Artificial Life, 6(3):219-226, 2000.

[8] G. Theraulaz, E. Bonabeau, S.C. Nicolis, R.V. Sole, V. Fourcassie, S. Blanco, R. Fournier, J.L. Joly, P. Fernandez, A. Grimal, P. Dalle, J.L. Deneubourg. Spatial patterns in ant colonies. Proc. Natl. Acad. Sci., 99(15):9645-9649, 2002.

[9] N.R. Franks and A.B. Sendova-Franks, Social resilience in individual worker ants and its role in the division of labour. Proc. R. Soc. London B, 256: 305-309,1994

[10] S.J. Backen and A.B. Sendova-Franks and N.R. Franks, Testing the limits of social resilience in ant colonies. Behavioral Ecology and Sociobiology, 48: 125-131, 2000.

[11] D. Grünbaum, Schooling as a strategy for taxis in a noisy environment. Evolutionary Ecology, 12: 503-522, 1998.

[12] M.D. Cox and G.B. Blanchard, Gaseous templates in ant nests. Journal of Theoretic Biology, 204:223-238, 2000.

Explicit von Neumann-style Reproduction and Genetic Relativism in Tierra

Axel von Kamp

Department of Bioinformatics
Friedrich-Schiller-University Jena
07743 Jena, Germany
kamp@minet.uni-jena.de

Abstract

A new type of Tierra ancestor which possesses an explicit genetic architecture is demonstrated. This ancestor exhibits genetic relativism in the form of a neutral drift of its genetic code. Although in this specific case the ancestor's evolvability is not increased, this example can serve as a starting point for considerations of the relevance of an explicit genetic architecture and genetic relativism for the evolvability of self-reproducing programs.

1 Introduction

Self-reproduction of programs in Tierra-like systems [7, 1, 11] is currently achieved by copying the sequence of instructions the program consists of. The procedure, which copies the instructions, is part of the program itself. Compared to the explicit genetic architecture of the von Neumann self-reproducer based on a universal constructor [6], one could at first sight come to the conclusion that programs in Tierra reproduce by self-inspection rather than through genetic reproduction. The reason why this distinction matters is that for open-ended evolution a genetic architecture is held to be desirable [10, 5].

As pointed out in [10], the programs in Tierra do indeed possess a genetic architecture, but there are two reasons why this can be overlooked: Firstly, the different terminology used for the description of the von Neumann genetic architecture and the operation of a typical Tierra program is a source of confusion. Secondly, the von Neumann self-reproducer consists of four distinct components all of which are embedded in a cellular automaton that represents the world, whereas in Tierra two of these components are usually largely implicit in the operation of the Tierra Simulator. Furthermore, the von Neumann self-reproducer is associated with a tape that carries the description of its four components.

The four components of the von Neumann self-reproducer have different functions: one is a universal constructor that interprets the tape and constructs the offspring according to this description, the second copies the tape, the third coordinates the actions of all the components and the fourth represents additional functionality not directly involved in the reproduction process. In Tierra, the instructions are represented by bit-strings and a global opcode-map defines the interpretation of bit-strings as instructions. The opcode-map is read when the Tierra Simulator is started, sets up the decoding function of the virtual CPUs and remains fixed during a run. Because the opcode-map is invertible, it is usually not necessary to point out the distinction between the instructions and the bit-strings representing them.[1] Therefore, when the sequence of instructions that constitutes the program is seen as the genotype, it plays this role only in conjunction with the opcode-map. Since the opcode-map is external to the Tierran programs and unchanging, this situation could be called genetic absolutism [5].

The opposite, genetic relativism, is clearly possible given the explicit genetic architecture of the von Neumann self-reproducer. Because the universal constructor is encoded on the tape, mutations in the section of the tape representing the constructor can potentially change the way the tape is interpreted. If such a mutation occurred, von Neumann saw it as unlikely that the offspring of the mutant would retain its self-reproduction capability [6, p. 86] and therefore discounted genetic relativism. But even if such a mutant could reproduce itself and genetic relativism should be a viable mutational pathway, it would still seem to be unnecessary for an evolutionary growth of complexity in the von Neumann self-reproducers.

The theoretical possibility of an evolutionary growth of complexity with the von Neumann self-reproducer relies critically on the independence of the fourth component from the reproduction process. Because of its independence, this component can represent additional functionality of any degree of complexity without interfering with self-reproduction. Some way or other, all these components of different complexity are connected by mutational pathways and it is reasonable to expect, that for a large number of components at least some mutational pathways always lead to a component of higher complexity. This makes an evolutionary growth of complexity in principle possible, but how it could be realized is a more complicated matter [5, sec. 5] outside the scope of the present article.

2 A different type of ancestor

It is important to distinguish between Tierra as a simulation platform and the original ancestor, as shown in [7, app. C], because it is possible to make the components of the von Neumann self-reproducer which are largely implicit in the original ancestor explicit in a different kind of ancestor which I call vn-ancestor. However, the component representing additional functionality is neither present in the original ancestor nor in the vn-ancestor. Nonetheless, because of its importance towards considerations about evolvability, it will be hypothetically discussed in the next section.

First of all, construction in Tierra is achieved by means of copying instructions from a source to a target position. In the original ancestor, the source is the sequence of instructions in the memory block belonging to the parent. This sequence is sequentially copied to the target which is the memory block allocated for the offspring. But what is copied is actually not the instruction but only the bit-string that represents the instruction under the opcode-map. To make the next conceptual step, copying can be seen as reading a bit-string from the source, passing it through an implicit identity function, and writing the result to the target.

Now one can easily conceive of an ancestor, which makes this identity function explicit by using a look-up table. Instead of copying the bit-string from the source to the target position, it uses the bit-string as an index for the look-up table and then copies the bit-string from the indexed entry to the target position. In the case of the identity function, the latter bit-string is the same as the one used as index. In principle, the look-up table can correspond to any possible function from bit-strings to bit-strings, but as long as the ancestor is involved in self-copying for reproduction, the identity function is the only possible one in order to construct offspring that is identical to its parent.

[1]Probably for this reason the underlying layer of bit-strings has been removed in Avida [1].

CREATURE BEGIN MARKER	EXECUTABLE					TABLE BEGIN MARKER	. . .
	calculate size	allocate memory	*interpret tape and construct offspring*	copy tape	divide		

. . .	TRANSLATION TABLE	TAPE BEGIN MARKER	TAPE		CREATURE END MARKER
			executable	translation table	

Figure 1: Architecture of the vn-ancestor. The three main parts are designated with capital letters and separated by markers written in small capitals. The division of the main parts into subparts with different functions is shown in normal writing and the subpart of the executable which belongs to the constructor is highlighted with italics.

This is the point at which the tape enters the picture. Instead of using the sequence of instructions that is executed by the virtual CPU as a source for copying, a dedicated tape can be attached to the ancestor which is used as source during construction of the offspring. At this stage, the vn-ancestor is complete. As shown in figure 1, it consists of three main parts: the executable, a look-up table which from now on will be called translation table, and the tape. This tape needs to be copied in a complete self-reproduction cycle, which can be achieved by an ordinary copy procedure like the one used in the original ancestor which there copies the sequence of instructions. If the translation table corresponds to the identity function, then for self-reproduction to take place the tape is just a duplicate of the executable plus translation table.

The architecture shown in figure 1 also makes it clear how the different components of the von Neumann self-reproducer are realized in the vn-ancestor: The subpart of the executable, that interprets the tape and constructs the offspring (i.e. the offspring's executable and translation table) together with the translation table corresponds to the universal constructor. The second component, which copies the tape, has its direct correspondent in the executable. Finally, the remaining subparts of the executable constitute the third component of the von Neumann self-reproducer and coordinate the actions of the other components.

Of course, now that there is a distinction between the executable and the tape, the translation table can correspond to almost any of the possible functions and achieve self-reproduction with an appropriately encoded tape. The only restriction is, that the range of this function must include all the bit-strings which under the opcode-map represent the different instructions used in the executable part of the vn-ancestor. This for instance rules out the constant functions each of which maps all bit-strings to one bit-string. The translation table is similar to the opcode-map in that it defines which bit-string on the tape represents which instruction. Thus, it allows the vn-ancestor to achieve independence of the global opcode-map as far as reproduction is concerned, whereas for the original ancestor the opcode-map is absolute. The tape, which is just a sequence of bit-strings, is now interpreted using the translation table.

It is possible to describe the architecture of the vn-ancestor in terms of the genetic architecture of biological cells in the following way: The translation table specifies the genetic code, the bit-strings on the tape correspond to the codons of the DNA/mRNA and the entries in the translation table to the tRNA's. The genetic architecture of the vn-ancestor is specified by the constructor subpart of the executable and the translation table represents an exchangeable

part	EXECU-	TRANSLATION		TAPE		
	TABLE	TABLE		executable	translation table	
position	P	M	N	P	M	N
start	I	x	I	N	y	N
mutation 1	I	I	I	N	N	N
mutation 2	I	I	I	M	N	N
mutation 3	I	I	x	M	N	y
chimera-1	I	I	I	N	y	N
chimera-2	I	I	x	N	y	N

Table 1: State of the creatures after the different mutations and the two chimeras; for explanation see text. x and y can be any bit-string as long as $x \neq I$ and $y \neq N$.

genetic code. Because in biological organisms the genetic code is universal[2], the sequence of nucleotides in the DNA is seen as the genotype, but for the vn-ancestor the tape can only be meaningfully seen as its genotype relative to the genetic code specified by the translation table.

3 Genetic relativism

Because the parts of the vn-ancestor which constitute the universal constructor are encoded on the tape, mutations in the corresponding segments of the tape will affect the construction mechanism found in the offspring. When the mutations affect the section of the tape that represents the constructor subpart, the genetic architecture could be changed. If the section of the tape representing the translation table mutates, the genetic code changes. This shows, that genetic relativism is in principle possible in Tierra. Although the original ancestors do not exhibit genetic relativism [5], it is not clear if they could evolve into creatures that show genetic relativism, but this seems to be quite unlikely. In any case, with the vn-ancestor genetic relativism can be realised, but in this case it takes the form of a neutral drift between different translation tables, all of which have essentially the same evolutionary potential.

It is possible to illustrate this drift with a thought experiment: First of all, the translation table must allow for some redundancy, so that the set of bit-strings that occur on the tape has fewer elements than the number of entries in the translation table. In this way, the table can contain unused entries. Now select an instruction, which only occurs once in the executable, called I. The position P on the tape, which encodes I, contains a binary number called N. Because I only occurs once in the executable, this N only occurs once on the tape. Accordingly, entry N of the translation table contains the instruction I (more precisely, the bit-string that represents I under the opcode-map). Now imagine a mutation that affects one of the unused entries of the translation table, say at index position M, so that it too contains I. A second mutation then transforms the value N at position P of the tape to M, which again is a unique number on the tape, because the corresponding table entry was previously unused. The third and last mutation affects the translation table again and changes the entry at index position N so that this entry does not contain I any more. Table 1 summarizes this sequence of events.

Now consider the creature after the second mutation and replace its tape with the one from the starting creature, creating chimera-1. The offspring of chimera-1 will be the starting creature, despite the fact that the translation table has been changed by the mutations. This backwards-compatibility is brought about by the redundancy in the translation table. After the third

[2]There are a few exceptions, for instance in the mitochondria of eukaryotes some of the codons represent different amino acids than they would in the nucleus. This shows, that within the genetic architecture used by biological cells different genetic codes are possible too.

mutation however, the corresponding chimera-2 will in its executable have the instruction I replaced by instruction x. Hence, the translation table is not compatible with the original tape anymore. If instruction I is a vital instruction like `divide`, which separates the offspring from the parent, the offspring of the chimera-2 will be sterile.

Why different vn-ancestors, despite being genetically incompatible, have essentially the same evolutionary potential can be illustrated by considering a hypothetical subpart of the executable called AF. This subpart would correspond to the component of the von Neumann self-reproducer that represents additional functionality and it would be encoded by its own section on the tape. The tape can be affected by point mutations which either add a random bit-string, delete one at a random position or replace one with an arbitrary bit-string. What I will consider as evolutionary potential here is the set of modified AFs reachable with any of the possible point mutations from a given AF. In the context of the vn-ancestor, these are point mutations affecting the corresponding section of the tape. Therefore, when an instruction is added to AF or one of them is replaced, then the new instruction can be only one of those that are in the range of the current translation table. When the translation tables of two incompatible vn-ancestors have the same range, then despite the circumstance, that the same mutation can have different effects with different translation tables, the set of modified AFs reachable is the same for both of them. Hence, their evolutionary potential is identical.

However, the preceding argument also shows, under which conditions the situation is changed so that the evolutionary potential is influenced: Firstly, the point mutations in Tierra can be configured to be either an arbitrary replacement or bit-flips. When they are restricted to bit-flips, the set of instructions that can be reached with a mutation which changes a particular instruction depends on how the translation table is organized. Therefore, the sets of AFs reachable from a given AF under an arbitrary point mutation are now different for every translation table, even when the ranges of the corresponding functions are the same. Secondly, it is obviously possible to have translation tables whose corresponding functions have different ranges and therefore can reach different sets of AFs.

Changes in the translation tables can be readily demonstrated with experiments in Tierra. The Tierra Simulator is configured so that only offspring which are of the same length as the parent are viable, and mutations are restricted to cosmic ray mutations, execution flaws and copy mutations, all of which result in replacements with arbitrary bit-strings. This helps to ensure that the overall architecture, as shown in figure 1 remains unchanged and simplifies interpretation. For the construction of a vn-ancestor, a random valid translation table is set up, according to which the executable and the table itself is encoded to produce the tape, and all three parts are spliced together. In order to have a high degree of redundancy in the translation table, it contains 128 entries whereas the executable only utilizes 30 distinct instructions.

As expected, neutral drift changes the translation tables over time and they become incompatible. By neutral drift is meant that the gestation time of the majority of genotypes that are extracted during a run remains largely unchanged. In addition to that and despite the restrictions mentioned above, changes in the executable appear over time as well, but the genetic architecture as such remains constant. This also holds when the mutations are restricted to bit-flips. When size changes are allowed and mutations include deletions, insertions and crossover, then in the experiments conducted so far, the vn-ancestor with its variants continues to dominate the populations without any changes in their lengths.

The neutral drift exhibited with the vn-ancestor so far does not enhance its evolutionary potential compared to the original self-copying ancestor, which becomes clear when one considers the hypothetical subpart AF which could be embedded in both the original and the vn-ancestor.

Firstly, it would in both cases be the same sequence of instructions. Also, to be sure that this part does not interfere with the reproduction process one could think of it as being executed before any of the parts associated with reproduction. Therefore, it is possible to compare the evolutionary potential of this hypothetical part being embedded in the original ancestor to being embedded in the vn-ancestor. If it is embedded in the original ancestor, then mutations affecting this subpart will be directly heritable whereas in the vn-ancestor one would have to consider analogous mutations of the tape's section that encodes this subpart. In the latter case, as discussed above, new instructions that can be added or replaced are those in the range of the translation table. Ideally, the range should then encompass all the instructions defined in the opcode-map. In this situation, the set of modified AFs reachable with point mutations that are arbitrary replacements is the same as if AF was embedded in the original ancestor, because in the latter case all instructions defined in the opcode-map can be accessed.[3] Hence, the evolutionary potential of the vn-ancestor is at best the same as that of the original one. For vn-ancestors with translation tables whose corresponding ranges contain less instructions than defined in the opcode-map, the evolutionary potential is accordingly decreased.

4 Other constructors

The considerations and experiments with the vn-ancestor have illustrated how genetic relativism can arise through a changing genetic code within an unchanging genetic architecture. In order to achieve a positive effect on the evolvability, one starting point could be to find a genetic architecture with the property that changes in the genetic code introduce neutral networks with similar characteristics to those in the space of RNA secondary structures [9]. These neutral networks interconnect clearly distinct molecule shapes through series of neutral mutations which would not be reachable with single non-neutral mutations. This is a different situation compared to the neutral drift exhibited by the vn-ancestor where neutral mutations only change the translation table without any effect on the evolutionary potential as long as the different translation tables have the same range.

This leads to the question of transitions between different genetic architectures, which use their genetic codes in different ways. Because the instruction set of Tierra supports universal computation [4], arbitrarily complex mappings between tape and executable are possible. Furthermore, the current situation in which the parent constructs the offspring could be completely changed so that for instance the offspring develops its own program in a way similar to a self-extracting archive. For these transitions to be possible, the whole constructor must be encoded in the genotype. However, whether such transitions are likely at all and to what degree they could enhance evolvability seems at present to be a matter of speculation.

The vn-ancestor-like genetic relativism within a constant system genetic architecture is also relevant to the field of genetic programming [2] as exemplified by automatically defined functions (ADF). Normally, the functions called in the evolving programs are global and unchangeable. The ADF however are local to each evolving program and their definitions can evolve too. Thus, they can serve to decompose the problem that is to be solved by the program into subproblems solved by each of the ADF.

Something similar could be achieved in Tierra when the universal constructor for instance uses production rules of some kind so that one symbol on the tape can stand for a block of instructions. This would mean that a point mutation on the tape can exchange sequences of instructions which would otherwise necessitate the implementation of more complex genetic operators like the cross-over that has been introduced with network Tierra [8]. However, to actually implement

[3]For this reason, replacement with an arbitrary instruction was introduced to Tierra where originally point mutations were bit-flips.

a program that can handle production rules in Tierra would be pretty cumbersome because of the primitive assembler language and limited resources in the virtual CPU. The use of non-trivial gentoype-phenotype mappings with inherent neutrality has already been demonstrated for evolving grammar-encoded plant models [12].

In order to further explore genetic relativism, it would seem advantageous to specify the genetic architecture, once a promising one has been selected, in some kind of modified Tierra Simulator and have each creature carry its own genetic code in the genotype. A specification of a genetic architecture does not necessarily mean that a transition to a different one will become impossible. The situation could be similar to the current Tierra Simulator, where the original ancestor uses the genetic architecture provided but a different ancestor could specify its own architecture of another sort. A modified simulator could for instance support additional instructions, which facilitate the parsing and application of production rules and simplify the implementation of a universal constructor based on such rules.

5 Outlook

The case for genetic relativism presented here is so far based on circumstantial evidence only. One approach for further investigation of neutral mutations would be to select one aspect of the evolving programs' phenotypes in relation to which neutrality can be defined, for instance the function performed by the subpart AF. Although this subpart is neither present in the original nor the vn-ancestor, it is part of the ancestor used in network Tierra [8] where it determines to which machine of the network the offspring is moved. It is also present in Avida [1] where it can evolve to solve tasks and gain additional CPU time for the reproduction process. For Avida, experiments have been carried out in which the fitness (replication efficiency) of the genotypes during the evolution of AF has been traced in detail [3]. Thereby it was shown that the fitness in the genealogy that leads to the final dominant genotype not always (although most of the time) increases. Not only are some mutations neutral, but fitness drops can also be observed. Detailed investigations of these deleterious mutations showed that they can be highly beneficial in conjunction with succeeding mutations. However, the organisms in Avida reproduce by copying themselves in the same manner as the original Tierra ancestor. It would be interesting to introduce a genetic architecture which supports genotype-phenotype mappings with different evolutionary potentials as well as neutral networks between those mappings in a Tierra-like system. Then the transitions between the different mappings and the role of neutral networks for these transitions during evolution could be investigated in detail.

6 Acknowledgements

For discussion of the topics in this article I would like to thank Tim Taylor and Barry McMullin. Financial support was provided by the Research Institute in Networks and Communications Engineering (RINCE) at Dublin City University.

References

[1] Adami C. and Brown C. T. (1994), Evolutionary Learning in the 2D Artificial Life System "Avida", *Artificial Life IV*, R. Brooks and P. Maes (eds.), MIT Press.

[2] Koza J. R. (1994), Genetic Programming II, MIT Press.

[3] Lenski, R.E. Ofria, C. Pennock, R.T. and Adami, C. (2003), The Evolutionary Origin of Complex Features, *Nature* 423, pp. 139-145.

[4] Maley C. C. (1994), The Computational Completeness of Ray's Tierran Assembly Language, *Artificial Life III*, C. Langton (ed.), Addison-Wesley.

[5] McMullin B. (2000), John von Neumann and the Evolutionary Growth of Complexity: Looking Backwards, Looking Forwards..., *Artificial Life VII*, M. Bedau, J. McCaskill, N. Packard, S. Rasmussen, MIT Press.

[6] von Neumann J. (1966), The Theory of Self-Reproducing Automata, University of Illinois Press.

[7] Ray T. S. (1991), An approach to the synthesis of life, *Artificial Life II*, C. Langton, C. Taylor, J. Farmer, S. Rasmussen (eds.), Addison-Wesley.

[8] Ray T.S. and Hart J. (1998), Evolution of Differentiated Multi-threaded Digital Organisms, *Artificial Life VI*, C. Adami et al. (eds.), MIT Press.

[9] Fontana W. and Schuster P. (1998), Continuity in Evolution: On the Nature of Transitions, *Science* 280, pp. 1451-1455.

[10] Taylor T. (1999), On Self-Reproduction and Evolvability, *Proceedings of the Fifth European Conference on Artificial Life (ECAL99)*, D. Floreano, J.-D. Nicoud, F. Mondada (eds.), Springer-Verlag.

[11] Taylor T. and Hallam J. (1997), Studying Evolution with Self-Replicating Computer Programs, *Proceedings of the Fourth European Conference on Artificial Life (ECAL97)*, P. Husbands and I. Harvey (eds.), MIT Press/Bradford Books.

[12] Toussaint M. (2003), Demonstrating the Evolution of Complex Genetic Representations: An Evolution of Artificial Plants, *Genetic and Evolutionary Computation Conference (GECCO 2003)*, pp. 86-97.

On bio-design of Argo-machine

Andrew Kuznetsov, Mark Schmitz, Kristian Müller

Institute of Biology III, Albert-Ludwig's University Freiburg,
Schaenzlestrasse 1, D-79104 Freiburg, Germany
andrei.kouznetsov@biologie.uni-freiburg.de
http://omnibus.uni-freiburg.de/~kouznet/

Abstract

We describe a non-deterministic abstract machine that searches according to oracle words in the design space to fit with its environment, cuts, transposes and pastes a set of tapes. On the basis of this construct a distributed concurrent computation with DNA by bio-molecules is proposed. For this purpose we are developing a site-guidable nuclease. Labeled oligonucleotides or PNA are used as an input. Two kinds of computation are discussed: 1) *in vitro* in a thermo-cycler or 2) *in vivo* after the transgene installation into living cells.

Key words: *biocomputing*, *guided nucleases*, *evolved systems*, *orthogonal life*

1. Introduction

A DNA based Turing machine had been proposed by Charles Bennet [1]; he used imaginary enzymes to perform the transition rules. Shapiro's group [2] for the first time developed an autonomous 2-state molecular automaton made of DNA, oligonucleotides, *FokI* endonuclease and ligase. Tom Head [3] introduced splicing systems (H-system, the formal model of DNA recombination) and demonstrated the computation on a plasmid by restriction enzymes and ligase [4]. Beaver [5] proposed that a universal Turing machine can be simulated by substitutions on DNA. Adleman [6] used the molecular biology tools for the solution of hard combinatorial optimization problems, i.e., the Directed Hamiltonian Path Problem. Landweber and Kari [7] showed the rearrangements of DNA and RNA, such as scrambling or editing, and proved the potential for solving computational problems that occur in biological systems. Recently we suggested the role of lateral gene transfer in the biological evolution [8] and compared it with the computation by communication [9]. The idea of a 'universal endonuclease' was proposed by Szybalski [10] and further developed in the Schultz's laboratory [11, 12]. They conceived that the catalytic domain of an endonuclease may be linked to a double- or triplex-forming oligonucleotide to generate an enzyme with novel specificities. In other reports to achieve a sequence-specific cleavage of DNA by topoisomerase I the guide oligonucleotide was covalently linked with the ligand to Topo I [13, 14].

Our investigation of computing by guided nucleases was inspired by recent data concerning Argonaute proteins and siRNA/RISC complex [15], and by the paradigm shift from algorithms to computation by interaction [16]. We develop CPST computational model (*cut-paste-select-transpose*) that is close to Eilenberg P-systems [17]. Our logical construct is a good agreement with 'cut-and-paste' and 'copy-and-paste' model of bacterial evolution [18]. In order to achieve a molecular implementation we are currently designing the nuclease, which sequence-specificity can be guided by the labeled oligonucleotides or PNA. This could be seen as a 'molecular head' of the computing machine. We consider two possibilities: 1) reaction in the test-tube in a thermo-cycler, or 2) cellular computing after an installation of the transgene coding the nuclease into living cells. A scheme is proposed and analyzed, where initial results allow establishing perspectives and project on more complex problem.

2. An abstraction

Consider an evolving system – an abstract machine and an environment that is continuously changing creates input words for the machine to stimulate an adaptation of this device to the surrounding.

Description. The *Argo-machine* (***AM***) consists of a finite set of *agents* (*Argonauts*); each of these has a head, a finite tape and can be in different states, which we specify as the output states[1]. The tape is a nonempty string of symbols that may be linear or circular. The head scans the tape according to an input word w_i, and cuts the tape at recognized sites. The agent arbitrarily pastes the tape's fragments[2]. For each tape-configuration there is an appropriate output state of the agent that is checked by the environment. Agents take special 'accept' and 'reject' actions. An agent accepts, if its output state corresponds to the environment state; an agent will reject if less than two matches to the input word exist on the tape. ***AM*** can accept if at least one agent accepts, reject if all agents reject, or loop. If the environment has changed, then it delivers a transposition[3] and a new word w_{i+1}. ***AM*** looks for an agreement with the environment permanently (Fig.1).

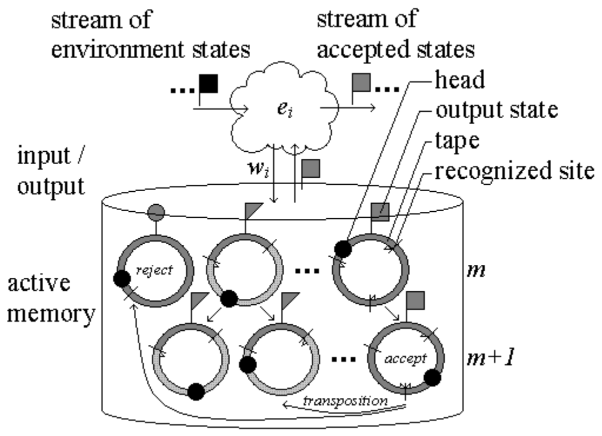


Fig.1 Schematic of *Argo-machine* (***AM***) with circular tapes.
The system operates on inputs and active memory, indirect uploads the memory and yields outputs (see text).

[1] The output state of the agent and the state of environment mean a string, a computable number (vector), a structure, e.g., the fractal [19] or a biological feature in applications. There is a mapping from the tape to the output state, from genotype to phenotype.
[2] The phrase "arbitrarily paste the tape" means to cut the tape into fragments and join them back together in an arbitrary order with or without the turn over. Fragments can not migrate between agents.
[3] The transposition means to make a copy of the tape from the accepted agent to other ones and join it in back-to-front.

In short, ***AM*** is a set of stochastic cut-paste agents, which act in parallel on their own tapes accordingly the instructions (input words), communicate with each other by transpositions of the tape and interact with the environment to compare the output states. Based on the comparison it accepts or runs in a loop to fitness to the environment. Each agent executes the same action, the *Argonaut's algorithm*.

Argonaut algorithm. $\mathscr{A}$ = "On word w:

1. Scan the tape to be sure that it contains at least two matches. If not, reject.
2. Cut at the matching sites and arbitrarily paste the tape's fragments.
3. Take the output state according the new tape.
4. Check it with the state of environment. If satisfy, accept; otherwise loop."

See Example 1 in the Appendix; here the states of the agent and an environment are represented by strings on the same alphabet, the tape is linear.

An Argo-machine is the 5-tuple system ***AM*** = (Σ,O,E,A,ζ), where

1. Σ is a finite alphabet, I and T are nonempty sets, such that I is the 'oracle' language over Σ and T is the set of finite strings (tapes) over Σ;
2. $O = \{ct,pt,sl,tn\}$ is a finite set of operators on T, here ct, pt, sl, tn denotes cut, paste, select and transpose (copy) respectively;
3. E is an infinite set of environment states;
4. A is a finite set of agents, such as $\boldsymbol{A} = (T,S,\delta)$, where

- S is a large finite set of output states, including sets: B, F, R, where B is the subset of initial states, F is the subset of final states, R is the subset of reject states;
- δ is the transition function, which according to the algorithm $\mathscr{A}$ for each word $w_i \in I$ at the input of agent $a_j \in A$ after operations $\{ct,pt\}$ on the tape $t_j \in T$ assigns an output state $s_j \in S$ by the mapping f: $T \rightarrow S$. The state s_j that is compared with the current environment state $e_i \in E$. Formally this can be expressed as $\delta(w,t) = (ct,pt,s,e)$;

5. ζ is the super-transition function, which generates the transposition $\{tn\}$ of the tape $t_j \in T$ from the agent $a_j \in A$ in the state $s_j \in F$ to other agents $a^+ \in A$ after the selection event $\{sl\}$, assigns the new initial states $n^+ \in B$, and finally delivers the new word w_{i+1}, or formally $\zeta(e,s) = (sl,tn,n^+,w)$.

A computation scenario of the *Argo-machine* is the shuffling of tapes from an initial set T_0 until an accepted output state is reached, at which the device halts temporarily and is not evolving. We define it as an *adaptation*. In general, the computation never ends, because the environment changes permanently. The event of environment change is called a *catastrophe* if it causes a super-transition. A progression of adaptations and catastrophes is defined as *evolution*. The intermediate computation is defined to be the result produced by ***AM***, which is the fit to the current environmental state on input stream and attached at the left site of the output stream of accepted states (Fig.1). Because it is essentially driven by selection and input words we are using synonyms such as: 'oracle', 'guide', or 'generative' for the input words.

Transpositions can be seen as the messages from the winner agent to the other ones. The size of tapes of the agents may grow beyond a limit. Additionally, we consider the output states of agents, as the messages to an environment. The environment is not passive and has the attributes: it deals with a stream of states, it monitors the adaptive fitness of agents, it can also produce new evolution rules or new oracle words in this senses. In the best case, these words create the solutions, and transpositions increase the size of tapes and a complexity (see Example 2, Appendix), else the system runs in a competitive regime. On the other extreme, the system rejects these words. There are two main questions. *What oracle words are optimal for a creative combinatorial design? What are the rules to form the oracle language, which generates the library of solutions*?

An analysis. To demonstrate how the *Argo-machine* works, let us assume the word w_i, which leads to brakes at matching sites x on the tape t_j, then a random paste of the tape. The number of rearrangements (with the turn over) for the single circular tape is described by the combinatorial formula *r(x)*:

$$\begin{cases} r_0 = r_1 = 0, \\ r_x = 2^x * (x-1)!; x \geq 2 \end{cases} \tag{1}$$

where r_x is the number of rearrangements, x is the number of matching sites on the tape. To illustrate a combinatorial power of expression (1) we compare the discrete function *r(x)* with functions 2^x, e^x and $x!$, see Fig.2.

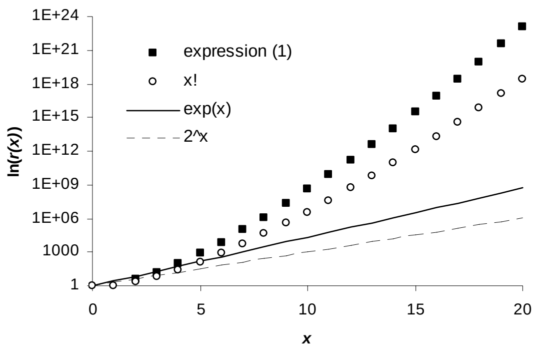


Fig.2 Combinatorial power of expression (1) versus 2^x, e^x, $x!$

Now consider the corresponding output state s that will be checked by the environment for each rearrangement r. According to the *Argonaut algorithm* $\mathscr{A}$, each agent a_j starts from the initial state s_j^0, runs on the oracle word w_i until the environment accepts at least one agent, then ***AM*** is stopped, Fig.3.

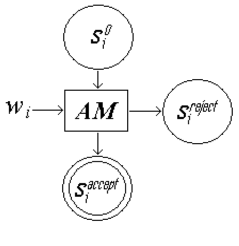


Fig.3 Evolving of *Argo-machine* from states s_j^0 to s_j^{accept} on word w_i ($s_j^0 \in B$, $s_j^{reject} \in R$, $s_j^{accept} \in F$, w_i is input word).

If the environment changes, then ***AM*** will require a new input. The catastrophe e_i generates the transposition tn_i, the winner agent concatenates a copy of its tape to another agents. After that the new initial states ${}_{i+1}s^{0}_{j}$ are formally assigned, the new word w_{i+1} starts up and the process runs ones more. If there is not any suitable word on input to satisfy the *Argonaut algorithm*, then all agents reject; now ***AM*** rejects and the process stops. We illustrate the evolution route on the example of winning agent's trace in Fig.4.

Computation is performed by a set of agents parallel in the nondeterministic manner. A computation power of this system depends on the number of agents and the number of output states for each agent. Note that it is impossible to run exponentially many agents simultaneously in real conditions; at least it needs a substitution of rejecting agents by winning ones. That is important the transpositions can propagate the local solutions and resolve some reject states. Transpositions enhance significantly the combinatorial-explosive search by increasing a dimension of the design space. This kind of distributed concurrent computation could permit a compositional design with the ruffle landscape. If a massive parallelism was achieved, then our model of computation would demonstrate a great computing power.

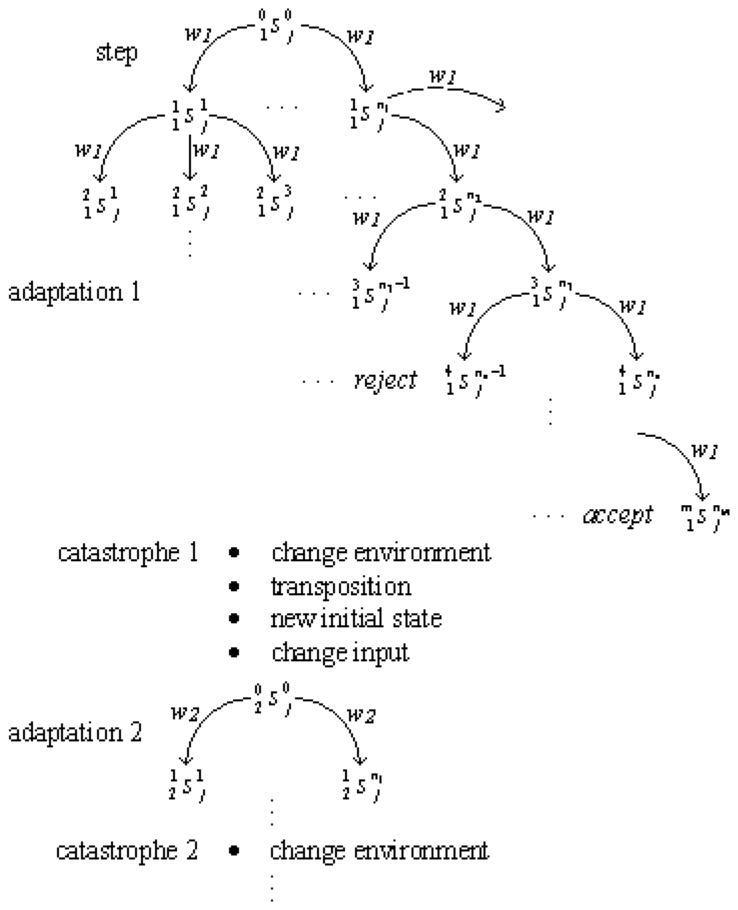


Fig.4 Nondeterministic computation by the agent *j*.
Here is ${}^{m}_{i}s^{n}_{j}$ - agent *j* in the state *n* on the step *m* before the catastrophe e_i, *i* - system's counter, w_i – oracle word.

3. Biological reality

We consider bio-molecules and living cells as suitable candidates to test the proposed *Argo-machine* in a real-world application. Besides, this molecular-scale machine may be a phenomenological model to understand the underlying biological processes.

Design. On the first step of investigation we created the fusion protein – IGNAF – comprising NucA non-specific endonuclease from cyanobacterium *Anabaena sp*. PCC 7120, and 4-4-20 scFv mouse single-chain antibody to fluorescein. Plasmids pBBInucA and pIG6Flullmh were used to construct pIGNucAFlu plasmid; pBBInucA was received from Alfred Pingoud (Giessen University), pIG6Flullmh was gifted by Andreas Pluckthun (Zurich University). *NucA* gene was cloned by PCR; a reverse primer encoded the sequence of tether. This cloned 780 bp fragment was inserted into pIG6Flullmh plasmid. As a result, IGNAF protein includes the ompA secretion signal, NucA domain, GSGGSGGSG peptide tether, variable light-chain (V_L) domain of scFv antibody, $(GGGGS)_6$ 30-mehr linker, variable heavy-chain (V_H) domain, and His-Tag. The C166G NucA mutant was obtained to avoid IGNAF dimer formation into the periplasma of *E.coli* cells. The additional mutations (R93A and W159A) were performed to decrease the nuclease activity and to avoid an inter-chain DNA cleavage. To increase the robustness of the enzyme two versions of protein are developed: (A) the monopod α-IGNAF is evolved by optimisation of NucA-domain via error prone PCR, shuffling and phage display, (B) the bipod β-IGNAF contains NucA split domain that is activated by self-assembling at the target DNA site. The split is located between β-sheet and α-helix in the T-P hinge region. Each part of NucA domain is linked to the own scFv antibody on N- and -C terminus accordingly (Fig.5). Thus, α-IGNAF has only a one ‘leg’ to contact to DNA by the specific oligonucleotide, although β-IGNAF has two ‘legs’ and theoretically is more robust.

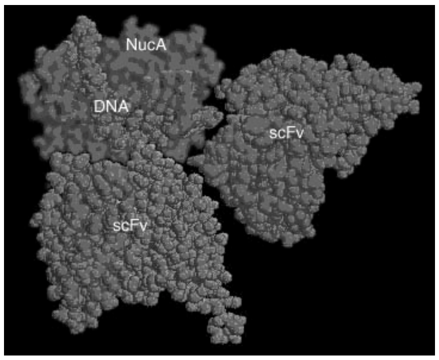


Fig.5 The model of β-version of IGNAF artificial nuclease on DNA.

Implementation. We suppose oligonucleotides or PNA labelled by fluorescein as *input-guide molecules*. IGNAF will bind to fluorescein and break DNA at the complementary sequence. We see two clearly distinguishable possibilities, the implementations *in vitro* (1) and *in vivo* (2).

1) The following alternatives *in vitro*:

a) the *catalytical approach* means that the nuclease is a catalytic with substrate turnover above the melting temperature T_m;

b) the *robust approach* will allow carrying out repeated hybridizations and cleavage reactions.

2) The required performances *in vivo*:

a) preinstallation of *ignaf*-transgene into living cells,

b) introduction of gene markers (*input-guide molecules*) in the cells,

c) activation of IGNAF nuclease at the target site.

Obviously IGNAF must specifically bind at a desired site, and then precisely cut the DNA chain in this region. Remarkably IGNAF is not an enzyme at the temperature less than T_m; it should be a genetic molecular device for applications *in vivo*, which acts only at the specific position. Smart IGNAF molecules have to bind at the target site, then switch on, next cleave DNA strand, and finally switch off. Thus under the control of guide molecules introduced into the system, computation will be performed in a loop, by the cleavage of the desired DNA pattern, and by a ligation to proceed the selected decision. Previous experiments demonstrated a limitation of specific cleavage by the ‘monopod’ guided nuclease [20]. To achieve the particular orientation of the nuclease on DNA the two different ‘legs’ would be more preferred. In this case the input comprising two half-words should be considered in the Argo-model. We do not discuss here a mechanism of transposition that could be implemented in the frame of the ‘minimal cell’ project. The compartmentalization of reactions is under the future investigations.

4. Conclusion

Computability and complexity are related to the development, adaptation, and evolution. Unfortunately, so far there is no computational definition of life; no minimal set of conditions needed for life to exist. This paper has been attempted to describe *adaptations*, *catastrophes* and *evolution* in computational terms. Following our research the requirements for a minimal life arise: How long should words and tapes be? What are the rules for the oracle words to effectively search by mapping in the design space? How many tapes and how much time does it require? Even under these constraints, our *Argo-machine* seems much like an oracle-choice evolved system, which performs an interactive ‘single-instruction, multiple-data’ computation in parallel. This concept combines *constructive*, *selective*, and *communicative* principles. This architecture could be applied to search the solutions of hard problems and to mimic the compositional evolution [21].

5. Acknowledgments

AK thanks Mikhail Kats for helpful discussion, Vladislav Erokhin for an implicit idea to use ‘cut-paste’ instead of ‘read-write’ operators, Alfred Pingoud and Andreas Pluckthun for DNAs, Alfred Sippel for a creative lab environment, and mother who told him about Argonauts. AK is solely responsible for any mistake herein.

6. Appendix

Argo-machine, a computation in the winning branch.

Language notations:

```
~,<,(,... – strings, cut before open brackets;
# - boundary symbol
```

Example 1. Adaptation without transposition:

```
environment '<~~>', word '<'
 1. <~~>                    environment
 2. <                       word
 3. #~<~<~<~#               tape_tick_1
 4. #~<~~><~#               tape_tick_2
 5. <~~>                    accept
```

Example 2. Two adaptations with one transposition:

```
environment_1 '<~(~>', word_1 '<',
environment_2 '<~~~>', word_2 '('
 1. <~(~>                   environment_1
 2. <                       word_1
 3. #~(<~<~)<~#             tape_tick_1.1
 4. #~(<~(~><~#             tape_tick_1.2
 5. <~(~>                   accept_1
 6. <~~~>                   environment_2
 7. #~(<~<~)<~##~(<~(~><~#  transposition
 8. (                       word_2
 9. #~(<~<~)<~~(<~(~><~#    tape_tick_2.1
10. #~(<~<~)<~~~>)(~><~#    tape_tick_2.2
11. <~~~>                   accept_2
```

7. References

1. Bennett C. H. (1982). The thermodynamics of computation. *International Journal of Theoretical Physics*, **21**, 905–940.
2. Benenson Y., Paz-Elizur T., Adar R., Keinan E., Livneh Z., Shapiro E. (2001). Programmable and autonomous computing machine made of biomolecules. *Nature*, **22(414)**, 430-434.
3. Head T. (1987). Formal language theory and DNA: an analysis of the generative capacity of specific recombinant behaviors. *Bulletin of Mathematical Biology*, **49(6)**, 737-759.
4. Head T., Rozenberg G., Bladergroen R. S., Breek C. K., Lommerse P. H., Spaink H. P. (2000). Computing with DNA by operating on plasmids. *BioSystems*, **57**, 87-93.
5. Beaver D. (1996). A Universal Molecular Computer. *DNA Based Computers (Selected papers from Proc. of DIMACS Workshop on DNA Based Computers'95), DIMACS Series in Discrete Mathematics and Theoretical Computer Science,* **27**, 29-36.
6. Adleman L. M. (1994). Molecular computation of solutions to combinatorial problems. *Science,* **266(5187)**, 1021-1024.
7. Landweber L. F., Kari L. (1999). The evolution of cellular computing: nature's solution to a computational problem. *Biosystems*, **52(1-3)**, 3-13.
8. Kuznetsov A. V., Kuznetsova I. V., Schit I. Yu. (2000). DNA interaction with rabbit sperm cells and its transfer into ova in vitro and in vivo. *Molecular Reproduction and Development*, **56(2)**, 292-297.

9. Kuznetsov A. V. (2004). The rules of sperm-mediated gene transfer. *Alife Mutants Hackingsession on Systems and Organisms (AMHSO), Rule 110 Winter Workshop, Bielefeld, Germany, 6-13 March.* Retrieved date, from http://www.rule110.org/amhso/results/gene-transfer.pdf
10. Szybalski W. (1985). Universal restriction endonucleases: designing novel cleavage specificities by combining adapter oligodeoxynucleotide and enzyme moieties. *Gene*, **40(2-3)**, 169-173.
11. Corey D. R., Schultz P. G. (1987). Generation of a hybrid sequence-specific single-stranded deoxyribonuclease. *Science*, **238(4832)**, 1401-1403.
12. Pei D., Corey D. R., Schultz P. G. (1990). Site-specific cleavage of duplex DNA by a semisynthetic nuclease via triple-helix formation. *Proc. Natl. Acad. Sci. USA*, **87**, 9858-9862.
13. Matteucci M., Lin K-Y., Huang T., Wagner R., Sternbach D. D., Mehrotra M., Besterman J. M. (1997). Sequence-specific targeting of duplex DNA using a camptothecin-triple helix forming oligonucleotide conjugate and topoisomerase I. *J. Am. Chem. Soc.*, **119**, 6939-6940.
14. Arimondo P. B., Bailly C., Boutorine A. S., Moreau P., Prudhomme M., Sun J. S., Garestier T., Helene C. (2001). Triple helix-forming oligonucleotides conjugated to indolocarbazole poisons direct topoisomerase I-mediated DNA cleavage to a specific site. *Bioconjug. Chem.*, **12(4)**, 501-509.
15. Parker J. S., Roe S. M., Barford D. (2005). Structural insights into mRNA recognition from a PIWI domain-siRNA guide complex. *Nature*, **434(7033)**, 663-666.
16. Wegner P, Goldin D. (2003). Computation Beyond Turing Machines. *Communications of the ACM*, **46(4)**, 100-102.
17. Gheorghe M., Holcombe M., Kefalas P. (2003). Eilenberg P systems: a bio-computational model. *Proc. First Balkan Conf. on Informatics, Thessaloniki, Greece,* 147-160.
18. Dawkins R. (2004). *The ancestor's tale: a pilgrimage to the dawn of evolution.* Boston: Houghton Mifflin.
19. Bentley P. J. (2004). Fractal Proteins. *Genetic Programming and Evolvable Machines Journal*, **5**, 71-101.
20. Corey D.R., Pei D., Schultz P.G. (1989) Generation of a catalytic sequence-specific hybrid DNase. *Biochemistry*, **28**, 8277-8286.
21. Watson R. A. (2006) Compositional Evolution: The Impact of Sex, Symbiosis, and Modularity on the Gradualist Framework of Evolution. (Vienna Series in Theoretical Biology) A Bradford Book. 324 p.

On the Evolution of Chemical Organizations

Naoki Matsumaru, Pietro Speroni di Fenizio, Florian Centler, and Peter Dittrich

Bio Systems Analysis Group
Jena Centre for Bioinformatics and
Department of Mathematics and Computer Science
Friedrich Schiller University Jena
D-07743 Jena, Germany
http://www.minet.uni-jena.de/csb

Abstract

Chemical evolution describes the first step in the development of life, such as the formation of complex organic molecules from simpler (in-)organic compounds. A deeper understanding of this period requires not only a refinement of our chemical knowledge but also improved theoretical concepts that help to explain how complex chemical systems evolve in principle. Here we investigate how chemical evolution appears in the light of chemical organization theory. We identify two main dimensions of chemical evolution: the "actual evolution" of the reaction vessel and the "organizational evolution" of the set of molecular species reachable from the actual set of chemical species present in the vessel. The organizational evolution can be described precisely as a movement through the set of chemical organizations. We describe three types of such movements: upwards, downwards, and sidewards. The concepts are illustrated by simulation studies on a constructive artificial chemistry.

1 Introduction

Chemical evolution (*i.e.*, prebiotic evolution) is concerned with the period of life's history that precedes the arrival of the first living organism [17]. Since Miller's pioneering work [19, 20], prebiotic chemistry has been studied in various laboratory experiments [16]. On the other hand, there are theoretical attempts to study chemical evolution. These theoretical approaches can be classified roughly into replicator centered and network centered approaches. The first approach assumes replicating molecules as the central unit of chemical evolution. Models like the Quasispecies [7] or *in-silico* RNA evolution [10] have characterized the capacity of chemical systems to store, transmit, and gain information.

The other line of research investigates how autocatalytic networks [7, 14, 21] emerge and evolve. An autocatalytic set can be defined as a set of molecules where each molecule is catalytically produced by at least one molecule from that set [12]. Therefore replication[1] like in the Hypercycle model [8] are not required for an autocatalytic set to maintain itself. It has been shown in silico that autocatalytic networks emerge under various conditions [9] and can possess complex dynamical properties [13].

[1]For example, a reaction $A + B + S \rightarrow 2A + B$ where A is replicated under the catalytic activity of B and by using up a substrate S.

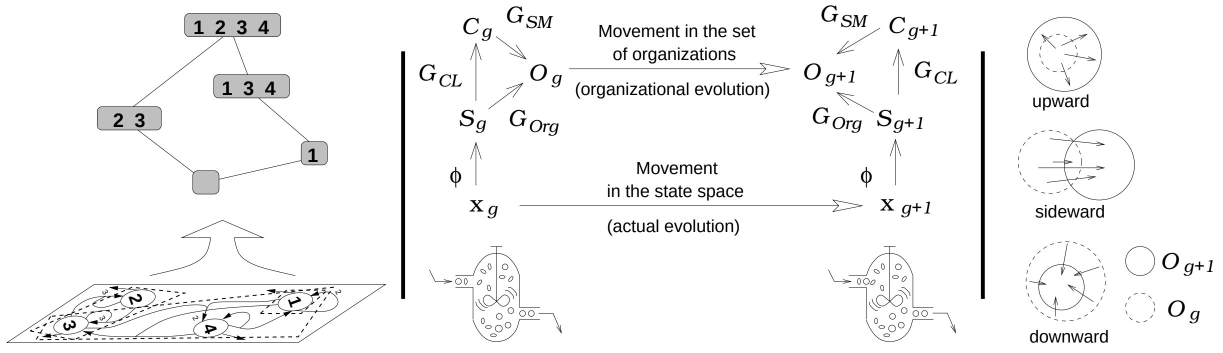


Figure 1: Schematic description of the static and dynamics analysis of a reaction system using chemical organization theory. **Left:** Based on the static network structure, the reaction network is decomposed into overlapping sub-networks called organizations. The hierarchical organizational structure of the network is visualized by a Hasse diagram. **Middle:** To analyze the system's dynamics, a movement from state $\mathbf{x}_g$ to a state $\mathbf{x}_{g+1}$ in state space is mapped to a movement from organization O_g to organization O_{g+1}. The actual evolution of the state $\mathbf{x}$ from g to $g+1$ does not necessarily lead to a change of the organizations, that is why we distinguish the *actual evolution* of the set of molecules actually present in the reactor from the *organizational evolution* of the organizations reachable from that molecules. **Right:** The organizational evolution is categorized into three movements: upward, downward, and sideward. See text for detail.

In order to study such complex dynamics new methods are required that can deal with constructive systems [11], *i.e.* systems where new components (molecular species) appear, which may change the present network topology. In this paper we study how chemical organization theory [22, 4] can help to explain the dynamics of constructive evolving chemical systems.

2 Chemical Organization Theory

The central concept of the theory is the definition of a chemical organization as a set of molecular species that is closed and self-maintaining [4, 11, 22]. In order to find the organizations of a reaction systems, the theory requires only the reaction network, which can be represented as an algebraic chemistry [5] without any dynamics. An algebraic chemistry is a 2-tuple $\langle \mathcal{M}, \mathcal{R} \rangle$ where $\mathcal{M}$ is a set of molecular species and $\mathcal{R}$ is a set of reaction rules among the species. A reaction rule describes a transformation of molecular species and can be represented as a pair of multisets of molecules, the left hand side and right hand side, respectively. Requiring only the algebraic structure, the reaction network is searched for the organizations. The first property — closure — ensures that there exists no reaction that produces new species not yet present in the organization using only species of that organization. The second property — self-maintenance — is a theoretical capability of an organization to maintain all of its members, (possibly) involving complex reaction pathways. Since the maintenance (possibly) involves complex reaction pathways, the stoichiometry of the whole reaction network must be considered, in general. Here, however, we investigate a specific class of reaction systems where all molecules are catalysts and where there is a general dilution flow. In this specific case, a set of molecules is self-maintaining if and only if every molecule within the set is produced by at least one reaction among molecules of that set.

By locating the organizations from all combinations of molecular species, the given reaction network is decomposed into overlapping sub-networks of organizations. As shown in Figure 1 (leftmost), we visualize the set of all organizations by a Hasse diagram, in which organizations are arranged vertically according to their size in terms of the number of their members. Two organizations are connected by a line if the upper organization contains all species of the lower organization and there is no other organization between them. The Hasse diagram represents the hierarchical organizational structure of the reaction network under study.

2.1 Dynamical Analysis

Chemical organizations are proposed to be an appropriate abstraction level to describe complex dynamical behaviors of reaction systems [22]. Central to this dynamical analysis is a function that maps a state $\mathbf{x} \in X$ of the reaction vessel to an organization generated by that state. Since the state space X is usually much larger than the set of all possible organizations (a subset of the power set of $\mathcal{M}$), this mapping provides a significant reduction of dimensionality. Given a state $\mathbf{x}$ (e.g., a concentration vector) we generate the organization in three steps:

First, the quantitative state $\mathbf{x}$ is mapped to a qualitative state $S = \phi(\mathbf{x})$, namely the set $S \subseteq \mathcal{M}$ of species present in $\mathbf{x}$. The function $\phi : X \mapsto \mathcal{P}(\mathcal{M})$ is called *abstraction*. Second, given the set of molecular species $S \subseteq \mathcal{M}$, we generate its closure $C = G_{CL}(S)$ by the algorithm shown in Tab. 1 (left). The closure of S is the smallest closed set containing S. Third, we generate the organization $O = G_{SM}(C)$ by finding the biggest self-maintaining set[2] ($O = G_{SM}(C)$) contained in the closure C (Tab. 1 (right)). All together the organization generated by a state $\mathbf{x}$ is defined as:

$$O = G_{SM}(G_{CL}(\phi(\mathbf{x}))) = G_{Org}(\phi(\mathbf{x})). \tag{1}$$

Given an algebraic chemistry $\langle \mathcal{M}, \mathcal{R} \rangle$ and dynamics of the reaction system as a movement in the state space X, the dynamical movement can be followed in the set of organizations L using Eq. (1) [4]. We call this movement *organizational evolution* in order to distinguish from the *actual evolution* of the state $\mathbf{x}$ of the reaction vessel (Figure 1). The dynamical movement on the organizational level can be categorized into three directions: upwards, downwards, and sidewards. Regarding two states $\mathbf{x}_{t_1}, \mathbf{x}_{t_2} \in X$ at time points t_1 and t_2, the organizations $O_{t_1}, O_{t_2} \in L$ can be generated: $O_{t_1} = G_{Org}(\phi(\mathbf{x}_{t_1}))$, $O_{t_2} = G_{Org}(\phi(\mathbf{x}_{t_2}))$. In case $O_{t_1} \supset O_{t_2}$, the movement in the state space is classified as a downward movement. The other way of inclusion, namely $O_{t_1} \subset O_{t_2}$, is an upward movement. The dynamical change ($O_{t_1} \neq O_{t_2}$) that is neither downwards nor upwards is called a sideward movement. We exclude the equality $O_{t_1} = O_{t_2}$, since no movement is detected on the level of organizations.

3 Organizations and Evolution

When we investigate the dynamical behavior of the reaction system on the level of the chemical organization, the dynamical behavior is categorized into there directional movements. Downward movement is ascribed mostly by the disappearance of a molecular species from the reaction vessel, consumed by internal reactions or decay. Upward movement, on the other hand, is brought by external perturbation such as mutation or insertion of new molecules. New molecular species are necessary to be produced for the system to move upwards. Furthermore, the new species must be impossible to be produced by chemical reactions among the species present in the reaction vessel because the closure generation function G_{CL} includes those species in the original organization.

Upward and downward movements are generally sufficient to describe the dynamical behavior. The other movement, sideward movement, occurs usually in a combination of the two movements. The system is excited by some external perturbation and triggers an upward movement. Then some species would disappear and cause a downward movement. If the series of movements results in an organization in which some of the original species are missing and some of the members are new, the movement is categorized as sidewards.

[2]The generated self-maintaining set is not always unique for a set of molecular species in arbitrary reaction systems, but is determined uniquely in the autocatalytic reaction system considered in this paper.

Table 1: Listing of functions to generate closed (left) or self-maintaining (right) set.

```
**Function:** Generate closure $G_{CL}$
**Input**: Set of species ($S$)
**Output**: Closed set of species ($CL$)

$CL \leftarrow S$
$A \leftarrow \emptyset$
while $A \neq \emptyset$ do
    $A \leftarrow \emptyset$
    foreach $(s_i, s_j) : s_i, s_j \in CL$ do
        $p \leftarrow s_i + s_j$
        if $p \notin CL$ then $A \leftarrow A \cup \{p\}$
    end
    $CL \leftarrow CL \cup A$
end
```

```
**Function:** Generate self-maintaining set $G_{SM}$
**Input**: Set of species ($S$)
**Output**: Self-maintaining set of species ($SM$)

$SM \leftarrow S$
$B \leftarrow SM$
while $SM \neq B$ do
    $B \leftarrow \emptyset$
    foreach $(s_i, s_j) : s_i, s_j \in SM$ do
        $p \leftarrow s_i + s_j$
        $B \leftarrow B \cup \{p\}$
    end
    $SM \leftarrow SM \cap B$
end
```

The upward and sideward movement are particularly significant in the field of evolution. When considering all sub-organizations in the reaction network, upward movement and sideward movement effect the new sub-organizations in the reaction network. Each (sub-)organization can be interpreted as a dynamical function of the reaction network [3], so the new sub-organization could be a new niche of the reaction system. By seeing the evolution of reaction network as the movement in the space of organizations, it could be practicable to analyze the functional evolution of the reaction system. The sideward movement is especially noteworthy since it captures the evolution without increasing the network size.

4 Experimental Setup

In this section, we demonstrate how the chemical organization theory gives an insight to chemical evolution. An artificial chemistry system called automata chemistry [6] is used to generate chemical evolution. Molecular species are binary strings $s \in \{0,1\}^{32}$ with a constant length of 32 bit. Two strings can catalyze the production of a third string: $s_1 + s_2 \Rightarrow s_3$. One of the strings s_1 is mapped to an automaton A_{s_1} according to a well defined instruction table (we used code table II in [6] allowing self-replication). The other s_2 serves as input to A_{s_1}. The result of the program execution on the input string is the product $s_3 = A_{s_1}(s_2)$. Preparing a reactor (or reaction vessel) containing N string objects, multiple copies of the species are placed in the reactor to simulate the dynamical behavior of the reaction system. In each time step, two string objects are randomly chosen to react, and the reactants are inserted back into the reactor without deleting the two reactands. One randomly chosen molecule in the reactor is replaced by the product in order to keep the total number of the objects in the reactor constant. In short, the system is a catalytic flow system in a well stirred reactor. In one *generation*, N steps are executed.

4.1 Analysis Method

Theoretically speaking, the automata chemistry consists of $2^{32} = |\mathcal{M}|$ binary strings as molecular species and reactions among them, forming the algebraic chemistry $\langle \mathcal{M}, \mathcal{R} \rangle$. Since it is impractical to consider the entire network, however, only the small part related to the reactor state $\mathbf{x}_g \in X$ at generation time g is considered as $\langle \mathcal{M}_g, \mathcal{R}_g \rangle$ where $\mathcal{M}_g = G_{CL}(\phi(\mathbf{x}_g))$ is the set of molecules that can be generated from $\mathbf{x}_g$.

Representing the reactor state as a multiset: $\mathbf{x}_g = \{m_1, m_2, \ldots, m_N\}$ where N is the size of the reactor, the abstraction of the reactor state is the set S_g of molecular species present in the reactor and calculated by ignoring the multiplicity: $S_g = \phi(\mathbf{x}_g) = \{s \in \mathbf{x}_g | \#(s \in \mathbf{x}_g) > 0\}$ where $\#(s \in \mathbf{x}_g)$ denotes the number of occurrences of element s in multiset $\mathbf{x}_g$. Taking the closure of the set of species: $C_g = G_{CL}(S_g)$ [23] listed as the pseudo code in Table 1 (left), the algebraic chemistry is constructed by setting $\mathcal{M}_g = C_g$. The set of reaction rules $\mathcal{R}_g = (\mathcal{C}_g \cup \mathcal{D}_g)$ is composed of two kinds of reactions: catalytic reactions $\mathcal{C}_g$ and decay reactions $\mathcal{D}_g$. Decay reactions $\mathcal{D}_g = \{(m \to \emptyset) | m \in \mathcal{M}_g\}$ are included since every object is subject to be replaced by a reaction product. In passing we note that it is not possible in the dynamical simulation of the reaction vessel to be empty even though every object species is defined to decay in the algebraic chemistry. Since two objects initiating a reaction are not altered by the reaction, the process is defined as a catalytic reaction $\mathcal{C}_g = \{(m_i + m_j \to m_i + m_j + m_k) | m_i, m_j \in \mathcal{M}_g, m_i + m_j \Rightarrow m_k$ according to the automata chemistry $\}$. Note that $m_k \in \mathcal{M}_g$ because of the closure property of the algebraic chemistry. As a result, there are $|\mathcal{R}_g| = |\mathcal{C}_g| + |\mathcal{D}_g| = |\mathcal{M}_g|^2 + |\mathcal{M}_g|$ reaction rules in the algebraic chemistry because the automata chemistry is designed to halt always by excluding the control statements.

Considering the characteristics of the reaction network, the function G_{SM} to generate the self-maintaining set can be defined as listed in Table 1 (right). The reaction network is designed so that every molecule decays but the reactants of all catalytic reactions are conserved. Therefore, the set is self-maintaining if all of the elements are produced by the catalytic reactions. In order to generate the biggest self-maintaining set contained in the original, species not produced by the reactions are excluded from the set until every molecule is produced.

Given the algebraic chemistry $\langle \mathcal{M}_g, \mathcal{R}_g \rangle$, we compute all organizations to extract the hierarchical organizational structure in the reaction network. The set of all organizations is denoted as L_g. It forms together with the union $\sqcup$ and intersection $\sqcap$ of organizations an algebraic structure $\langle L_g, \sqcup, \sqcap \rangle$ called a lattice[3]. The biggest organization $O_g \in L_g$ is generated from the whole set of the species present in the reaction vessel: $O_g = G_{Org}(S_g)$. The other sub-organizations are generated from any subset of the set: $O'_g = G_{Org}(S'_g \subset S_g)$

5 Results

In order to study down movements, we simulate our artificial chemistry without any external perturbations like mutation (Sec. 5.1). Then, in Sec. 5.2, we will demonstrate upward and side movements by introducing moderate mutations, which cause constructive perturbations.

5.1 Dynamical behavior as downward movement

The reactor of size $N = 1000$ is heterogeneously initialized with N random objects. Figure 2 shows the typical dynamical behavior of the simulated chemical evolution in three forms. The concentration change of some species (with relatively high quantity) is plotted to view the dynamical change of the reactor state. As in the middle graph, the number of molecular species present in the reactor is plotted as diversity. A tendency to decrease diversity may describe this evolutionary behavior. This dynamical behavior is analyzed with the theory of chemical organization, and the results are given as a series of Hasse diagrams, visualizing the lattice of organizations L_{200}, L_{400}, L_{500}, and L_{700} (Figure 2, bottom). The labels in the box indicate species that are new in the corresponding organization and are not contained in any of the organizations below it. Since the organizational structure depends on the qualitative state of the reaction vessel the result is not affected as long as the diversity stays the same.

[3]Since the algebraic chemistry is designed as a reactive flow system, the set of all organizations in such a system is proved [4] to form a lattice.

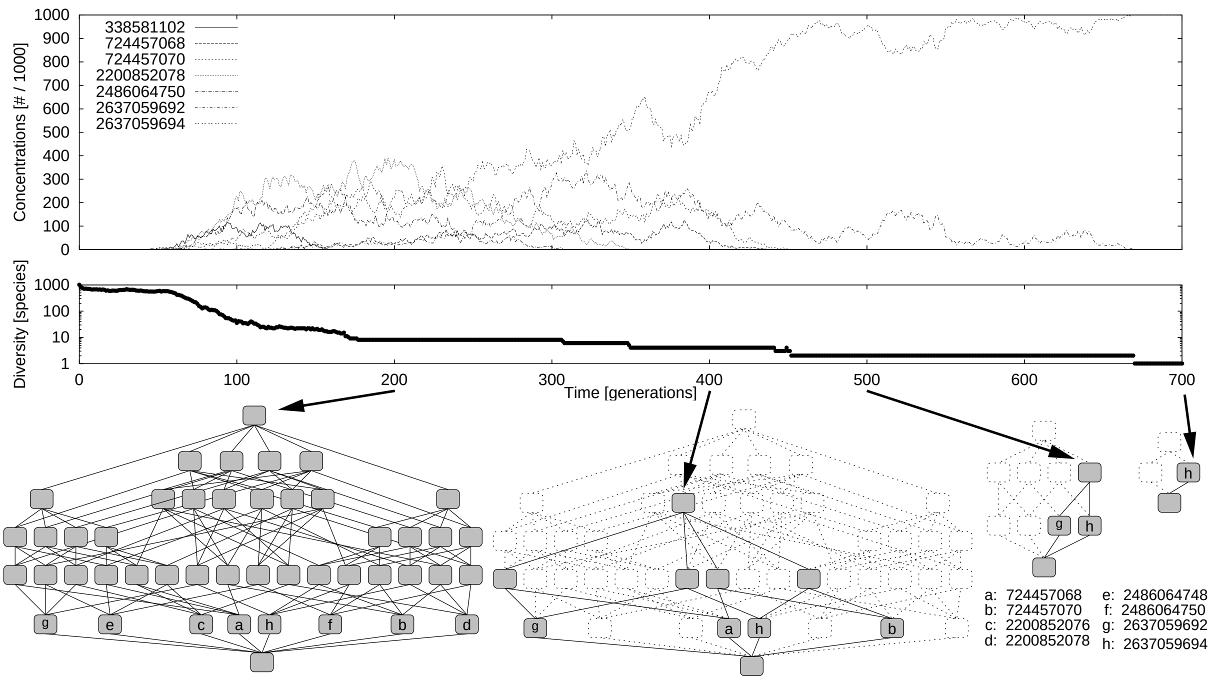


Figure 2: Dynamical behavior of automata chemistry showing several downward movements. The reactor of size $N = 1000$ is filled initially with one copy each of N species of random binary string with fixed length 32 bits. Top: concentration profile of the reactor with respect to some prominent species. Middle: diversity as the number of different species present in the reactor. Bottom: lattice of organizations at generation 200, 400, 500, and 700. Dotted boxes and lines are the organizations and links missing compared with the previous lattice structure.

At $g = 200$, there are eight species in the reactor, and those species form the biggest organizations: $O_{200} = G_{Org}(S_{200}) = S_{200}$. Forty-six organizations are found as shown in the leftmost Hasse diagram. In the next 200 generations, four species are drained so that the diversity value of the reactor becomes four. The lattice of organizations L_{400} consists of ten organizations including the biggest organizations formed by the remaining four species. Comparing two sets of the organizations L_{200} and L_{400}, we found that $L_{200} \subset L_{400}$ so that it is possible to impose the lattice L_{400} on L_{200} as shown in the figure. The solid lines represent the lattice at $g = 400$, and the organizations vanished during the 200 generations are drawn by the dotted lines

This dynamical change of the reactor state can be explained as a downward movement. The sets of species present in the reactor at generation $g = 200$ and 400 are the organizations: $O_{200} = G_{Org}(S_{200}) = S_{200}$ and $O_{400} = G_{Org}(S_{400}) = S_{400}$. The inclusion $O_{400} \subset O_{200}$ is true since species present in the reactor only disappear and no new species appears within that 200 generations. Similarly, this argument is applicable between S_{400} and S_{500} and between S_{500} and S_{700}. In this simulation settings, only the reactions can produce possibly new species, but applying the chemical reactions to the set of existing species cannot disrupt the closure property of the organization. Only the downward movement is thus feasible.

5.2 Upward and sideward movement

To demonstrate upward and sideward movement, a mutation process is introduced. Every 100 generation, ten objects are chosen randomly, and each binary string object is mutated by inverting one randomly chosen bit. The reactor is initialized homogeneously with $N = 1000$ copies of a certain species, so the diversity value is 1 in the beginning. Figure 3 (top) shows the

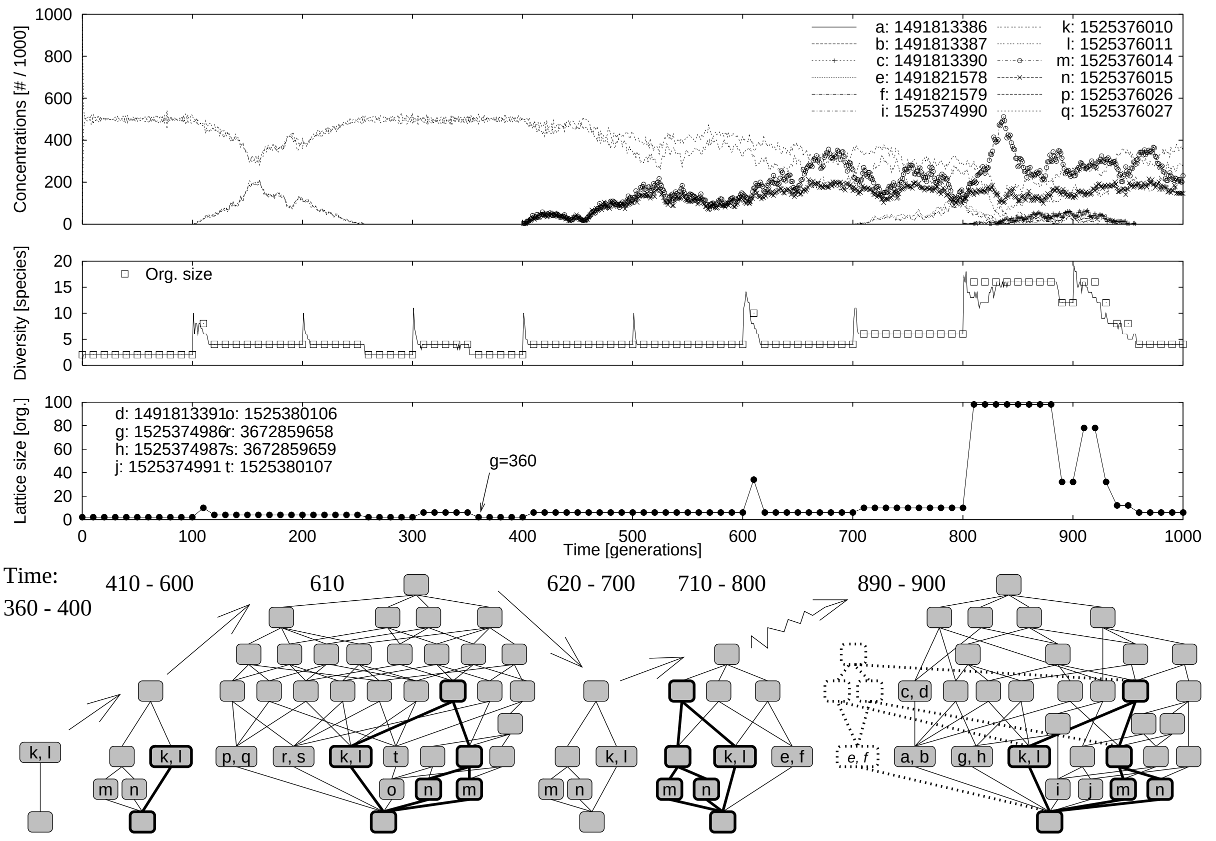


Figure 3: Dynamical behavior of automata chemistry exhibiting upward and sideward movement. The reactor of size $N = 1000$ is initialized heterogeneously. Every 100 generations, 10 string objects are chosen to be mutated by an one-bit negation. Top: dynamical change of concentration profile of the reactor with respect to prominent species. Second top: the number of unique species in the reactor as diversity and the size of the biggest organization generated, calculated every 10 generations. Third top: the number of the organizations in the reaction network. Bottom: Hasse diagrams depicting organizational structure in the reaction network. Starting from $g = 360$, two upward movements are achieved until $g = 610$ although the lattice immediately shrinks (downward movement). From 800 to 900, a sideward movement is observed.

dynamical behavior of the concentration profile with respect to the prominent species, and the number of the species existing in the reactor is plotted as diversity. The rapid increase of the diversity every 100 generation is caused by the ten new mutants. The organizational structure in the reaction network is computed every 10 generation. At the moment of the mutation event, the network is analyzed just before the mutation, and the effect of the mutation is observed only after ten generations. At the bottom, the dynamical change of the lattice structure from $g = 360$ is depicted. The organizations and links are drawn by bold lines if inherited from the previous structure, and the dotted lines are used if vanished.

Starting with two organizations (empty set and set of two species), the mutation at $g = 400$ introduces new species to the reaction system and the reaction network is expanded. After ten generations, the reaction system settles to the state with four species. The reaction network with the four species is composed of six organizations, and the biggest organization is the set of those four species. This lattice structure is sustained in the next 200 generations including one mutation at $g = 500$. Temporarily, the next mutation at $g = 600$ brings up the system to the organization of ten species and thirty-four organizations in the reaction network as observed at $g = 610$. After 20 generations, all of the new organizations are vanished, and the lattice structure comes back to that prior to the mutation at $g = 600$.

These are typical upward and downward movements. The mutation process produces new species outside of the closure and causes upward movement. The dilution flow removes species from the reaction vessel randomly, and the system goes to the organization below. Since the concentration of the new species is very low, the new organizations brought about by the upward movement has a disadvantage statistically. Thus, the upward movement is canceled mostly.

The sideward movement is observed between O_{800} and O_{900}. The mutation process at $g = 800$ introduces new species to the reactor system and pushes the system into the bigger organization consisting of sixteen species. Ninety-eight organizations are found in the reaction network. Each of the sixteen species is maintained for a relatively long period (90 generations), but four of the species are eventually depleted. In consequence, lattice structure L_{900} has 32 organizations. As illustrated in Figure 3 bottom, four organizations associated with species e and f are missing in L_{900} in comparison with L_{800}.

5.3 Diversity and Organization

In the previous examples, the generated organization from the reactor state is mostly the same as the abstraction (i.e., $G_{Org}(\phi(\mathbf{x})) = \phi(\mathbf{x})$). In other words, the set of species present in the reactor is an organization. In that case the diversity (number of different species present) seems an adequate representation of the evolutionary behavior. However, the benefit to apply the generate function becomes evident in Figure 4, where we can see that a decrease in diversity does not necessarily imply a decrease of the generated organization.

For this simulation the reactor of size $N = 1000$ is initialized with sixteen species, which form a reaction network holding 146 organizations as shown in Figure 5 (left). Mutation is disabled so that the only downward movement can occur. The reactor is in the biggest organization at the top of the lattice structure, and there are four organizations directly below as shown in Figure 5 (right). The organizational structure is sustained for a long time ≈ 800 generations until a series of species destruction causes the reaction system to move downwards. Around generation 200, the diversity is reduced due to the disappearance of the species from the reactor, and the reduction is dynamically compensated by regenerating the disappeared species. When the set of species present in the reactor is not the organization anymore, violating the closure property in this case, the dynamical reaction system tends to move the state so as to satisfy the two

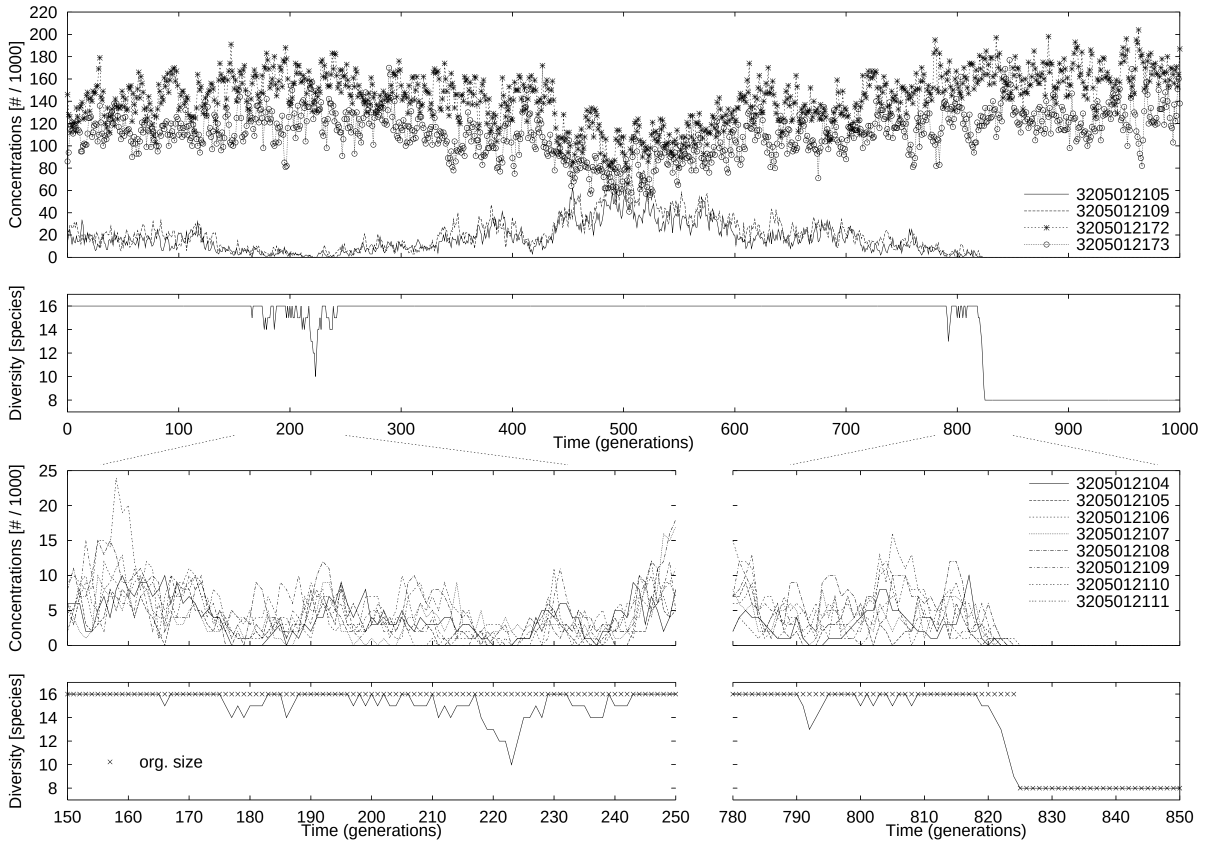


Figure 4: Dynamical behavior of automata chemistry exhibiting long-term preservation of an organization and then downward movement. The reactor of size $N = 1000$ is initialized with sixteen species, and the organizational structure in the reaction network among those species consists of 146 organizations as shown in Figure 5 (left). Top: concentration profile of the reactor with respect to the prominent species and diversity as the number of unique species present in the reactor. Bottom left: zoomed into [150:250] to show in detail the dynamical behavior compensating qualitative disturbance. Bottom right: zoomed into [780:850] where stochastic effects eventually caused downward movement.

properties of the organization since the organization is a candidate of the steady state and the other species combinations are not stable [4].

The organization generated from the reduced set of species is, however, unchanged during that period. Applying the generate function takes the structure of the reaction network into consideration. By representing the dynamical behavior on the level of the organization, the dynamical change of the underlying reaction network is focused. Furthermore, temporal stochastic effects can be separated from the permanent effects, causing downward movements.

6 Discussion and Conclusion

In this paper, we have demonstrated that chemical organization theory can provide another level of explanation to understand chemical evolution. With the help of the theory, we can consider two levels of chemical evolution: (1) the actual evolution of the reaction vessel, that is, the arrival and disappearance of chemical species; and (2) the "organizational evolution", that is, the change of the organization generated by the current set of molecules present in the vessel (Figure 1, middle). Our results suggest that actual evolution of the reaction vessel does not trivially imply its organizational evolution and vice versa.

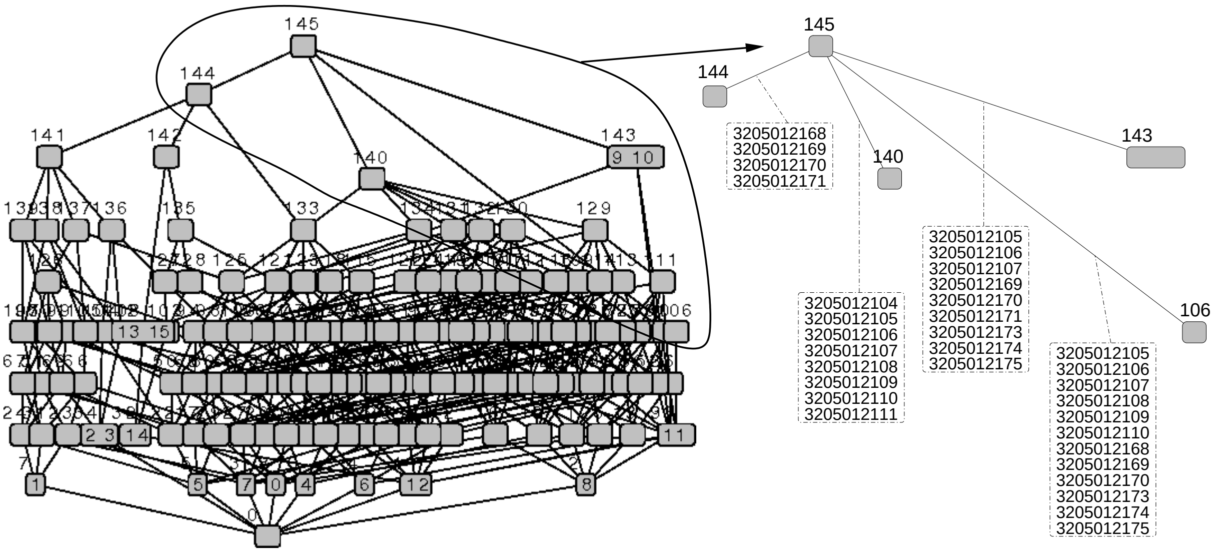


Figure 5: Organizational structure in the reaction network of the sixteen species with which the reactor for Figure 4 is initialized. **Left:** the whole lattice structure containing 146 organizations. **Right:** four organizations directly below the biggest organization (labeled as 145). For each link to below, the missing species are listed. Since twelve species constitute the organization labeled as 144, for instance, the link to that organization is associated with four species. The downward movement demonstrated in Figure 4 is from organization 145 to 140.

We have characterized, as usual in evolution theory, the actual evolution of the reactor by the change of its diversity, which reflects the arrival and disappearance of chemical species. The evolution on the organizational level was characterized as downward, upward, or sideward movement in the organization space. As suggested by our experimental results (Figures 2-4), downward movement correlates with decreasing diversity whereas an upward movement correlates with increasing diversity. However, in general, the relation between the actual level (actual state of the system) and the organizational level (organization the system is in) is not that simple. In fact, we have shown that there can be a decrease or increase in diversity without any change on the organizational level (*i.e.*, the organization generated does not change). Even a process that appears like a creative evolutionary process on the actual level can in fact be just a downward movement on the organizational level (see e.g. Figures 6 and 7 in Ref. [6]). In other words, an increase in diversity or the appearance of new molecular species (on the actual level) does not necessarily imply an upward or sideward movement but can go hand-in-hand with a downward movement (on the organizational level). Finally, it is even possible that an upward movement is accompanied by a decrease of diversity, *e.g.*, in case some new molecular species take a large portion of the reaction vessel, although we have not experimentally demonstrated this case, yet.

An important aspect left for future research is to characterize the intrinsic stability of organizations. As we observed, not all organizations show the same level of stability: some organizations are sustained over very long periods while others are inherently unstable, or unstable under the smallest external noise. What exactly makes an organization stable or unstable is at the moment only a speculation, yet the topology of the reaction network [24, 15] and the existence of an attractor inside the organization could be important aspects to take into account.

When investigating evolutionary processes, the issue of complexity is inevitable and controversial. Previous studies suggest that evolution shows unlimited growth of complexity [1]. According to a recent analysis [18], the research about machines that grow in complexity can be traced back to works by John von Neumann or even further to western philosophical and theological thinking. Fontana and Buss [11] presented an artificial chemistry in which systems' complexity would increase by combining multiple non-complex systems. Another example, a natural one, are biochemical signaling pathways which are coupled and display emergent behaviors, such as bistability [2]. We speculate that the organizational structure of the reaction network (the lattice structure of organizations) has a close relation to the complexity of the dynamical reaction system. Where the number of organizations in the network is a facet of the system complexity, because of the association between sub-organizations and dynamical functions. However the structural features of the lattice, not only the size of the organizational structure, must also be taken into consideration. Do the sub-organizations contain each other like a chain, or do they form a hypercube or even more complex patterns. All these aspects are obviously relevant to the evolution of the system, yet the exact way in which they would affect it are still to be investigated and evaluated.

Acknowledgments The authors would like to thank the members of our group for their support and useful advice. The acknowledgment also goes to Gerd Grünert and Christoph Kaleta for their help with the software development. This article is based upon work supported by the BMBF (Federal Ministry of Education and Research, Germany) Grant 0312704A. and by the DFG German Research Foundation Grant Di 852/4-1.

References

[1] M. Bedau and C. T. Brown. Visualizing evolutionary activity of genotypes. *Artif. Life*, 5:17–35, 1999.

[2] U. S. Bhalla and R. Iyengar. Emergent properties of networks of biological signaling pathways. *Science*, 283:381–387, 1999.

[3] F. Centler, P. Speroni di Fenizio, N. Matsumaru, and P. Dittrich. Chemical organization in the central sugar metabolism of escherichia coli. *Modeling and Simulation in Science, Engineering and Technology*, 2006. (accepted).

[4] P. Dittrich and P. Speroni di Fenizio. Chemical organization theory. *Bull. Math. Biol.*, 2006. (accepted).

[5] P. Dittrich, J. Ziegler, and W. Banzhaf. Artificial chemistries - a review. *Artif. Life*, 7(3):225–275, 2001.

[6] P. Dittrich and W. Banzhaf. Self-evolution in a constructive binary string system. *Artif. Life*, 4(2):203–220, 1998.

[7] M. Eigen. Selforganization of matter and the evolution of biological macromolecules. *Naturwissenschaften*, 58(10):465–523, 1971.

[8] M. Eigen and P. Schuster. The hypercycle: a principle of natural self-organisation, part A. *Naturwissenschaften*, 64(11):541–565, 1977.

[9] J. D. Farmer, S. A. Kauffman, and N. H. Packard Autocatalytic replication of polymers. *Physica D*, 22:50–67, 1986.

[10] W. Fontana and P. Schuster. Continuity in evolution: On the nature of transitions. *Science*, 280:1451–1455, 1998.

[11] W. Fontana and L. W. Buss 'The arrival of the fittest': Toward a theory of biological organization. *Bull. Math. Biol.*, 56:1–64, 1994.

[12] S. Jain and S. Krishna. Autocatalytic sets and the growth of complexity in an evolutionary model. *Phys. Rev. Lett.*, 81(25):5684–5687, 1998.

[13] S. Jain and S. Krishna. A model for the emergence of cooperation, interdependence, and structure in evolving networks. *Proc. Natl. Acad. Sci. U. S. A.*, 98(2):543–547, 2001.

[14] S. A. Kauffman. Cellular homeostasis, epigenesis and replication in radomly aggregated macromolecular systems. *J. Cybernetics*, 1(71-96), 1971.

[15] K. Klemm and S. Bornholdt. Topology of biological networks and reliability of information processing. *Proc. Natl. Acad. Sci. U.S.A.*, 102(51):18414–9, 2005.

[16] A. Lazcano and J. L. Bada. The 1953 Stanley L. Miller experiment: fifty years of prebiotic organic chemistry. *Orig Life Evol Biosph*, 33(3):235–42, 2003.

[17] J. Maynard Smith and E. Szathmáry. *The Major Transitions in Evolution.* Oxford University Press, New York, 1995.

[18] B. McMullin. John von Neumann and the evolutionary growth of complexity: Looking backwards, looking forwards... In M. A. Bedau, J. S. McCaskill, N. H. Packard, and S. Rasmussen, editors, *Artificial Life VII.* MIT Press, Cambridge, MA, 2000.

[19] S. L. Miller. A production of amino acids under possible primitive earth conditions. *Science*, 117(3046):528–9, 1953.

[20] S. L. Miller and H. C. Urey. Organic compound synthesis on the primitive earth. *Science*, 130(3370):245–51, 1959.

[21] O. E. Rossler. A system theoretic model for biogenesis. *Z. Naturforsch. B*, 26(8):741–6, 1971.

[22] P. Speroni di Fenizio and P. Dittrich. Artificial chemistry's global dynamics. movement in the lattice of organisation. *The Journal of Three Dimensional Images*, 16(4):160–163, 2002.

[23] P. Speroni di Fenizio, P. Dittrich, J. Ziegler, and W. Banzhaf. Towards a theory of organizations. In *German Workshop on Artificial Life (GWAL 2000), in print*, Bayreuth, 5.-7. April, 2000, 2000.

[24] J. Stelling, U. Sauer, Z. Szallasi, F. J. Doyle, and J. Doyle. Robustness of cellular functions. *Cell*, 118:675–685, 2004.

Author Index

Artmann, Stefan, 1
Ay, Nihat, 9

Bedau, Mark, 6
Bertschinger, Nils, 9
Beurton-Aimar, Marie, 85
Buck, Moritz, 51

Centler, Florian, 135

Dittrich, Peter, 1, 135

Gunji, Yukio-Pegio, 39

Jain, Sanjay, 7
Jost, Jürgen, 9

Kim, Jan. T., 63
Knabe, Johannes F., 75
Kuznetsov, Andrew, 125

Lales, Charles, 85

Magg, Sven, 95
Matsumaru, Naoki, 135
Mazat, Jean-Pierre, 85
Merkle, Daniel, 105
Middendorf, Martin, 105
Miura, Tomoya, 21
Müller, Kristian, 125

Nehaniv, Chrystopher L., 51, 75
Nomura, Tatsuya, 31

Olbrich, Eckehard, 9

Parisey, Nicolas, 85

Scheidler, Alexander, 105
Schilstra, Maria J., 75
Schmidhuber, Jürgen, 8
Schmitz, Mark, 125
Speroni di Fenizio, Pietro, 135

te Boekhorst, Rene, 95
Tominaga, Kazuto, 21
Tsuda, Soichiro, 39

von Kamp, Axel, 117

Zauner, Klaus-Peter, 39